OM

Sujal N. Patel ([redacted])
9574234622

ECE

PM1(B)

ACE

E. C. E

Dr. Chakrapani

Advanced Books	Text Books
① Samuel C. Lee	① Morris Mano.
② Kohavi	② Tocci & Woolmer.
③ Bogart	③ R.P. Jain

Digital Electronics

* Basic Topics

① Logic gates

② Number System

③ Complementary Number representation and Binary Number.

④ Binary codes.

⑤ Boolean Algebra.

⑥ k-maps.

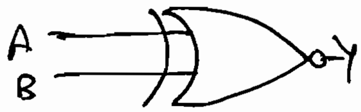
D: 23/6/2013

☆ Logic Gate:

→ AND, OR, NAND, NOR, EX-OR, EX-NOR gates.

→ NAND & NOR are Universal gates.

① Equavalance | coincidence gate
⇒ ExNOR gate

→ 

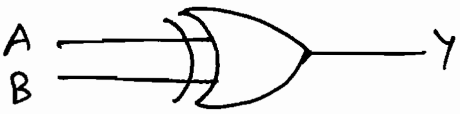
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	1

$$\boxed{Y = A \odot B}$$

$$\therefore \boxed{Y = \bar{A}\cdot\bar{B} + A\cdot B}$$

② Staircase Connection :-
⇒ Ex-OR gate Logic

→

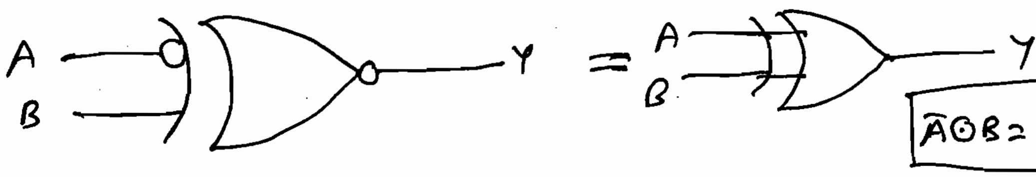


A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

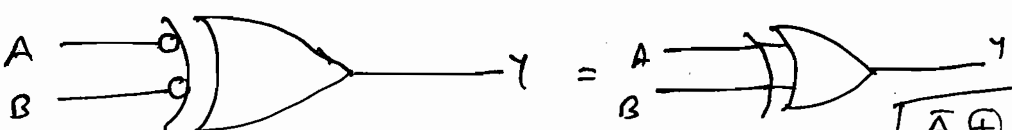
$$\boxed{Y = A \oplus B}$$

$$\boxed{Y = \bar{A}\cdot B + \bar{B}\cdot\bar{A}}$$

③

ⓐ 

$$\boxed{\bar{A}\odot B = A \oplus B}$$

ⓑ 

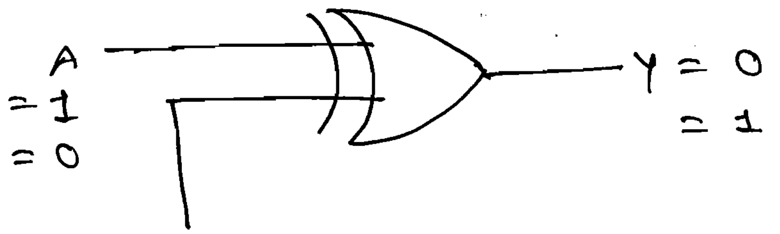
$$\boxed{\bar{A}\oplus\bar{B} = A \oplus B}$$

→ $\boxed{A \oplus \bar{B} = A \odot B}$

(4) <u>Min no. of gates</u>

	NAND	NOR
Ex-OR	4	5
EX-NOR	5	4

(5)

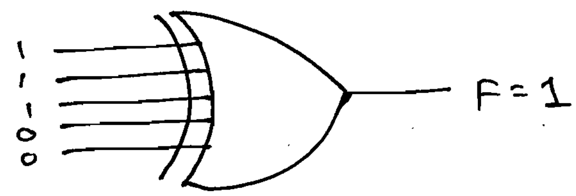


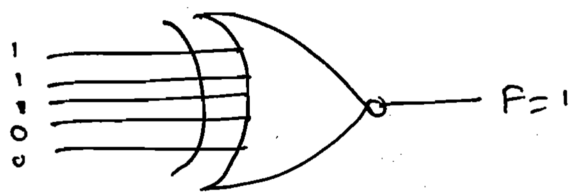
Control If $x = 1 \rightarrow Y = \bar{A}$ (Inverter)

'x' = 1 $x = 0 \rightarrow Y = A$ (Buffer).

NOTE: For X-NOR it reverse.

(6)





$1 \oplus 1 \oplus 1 \oplus 0 \oplus 0 = 1$

$1 \odot 1 \odot 1 \odot 0 \odot 0 = 1.$

NOTE:

→ Ex-OR output = 1 if Input has odd no of 1's

→ Ex-NOR Output = 1 if Input has Even no. of 0's.

→ $\boxed{\text{Ex-NOR} = \text{Ex-OR}}$ if no. of Input Variables are odd.

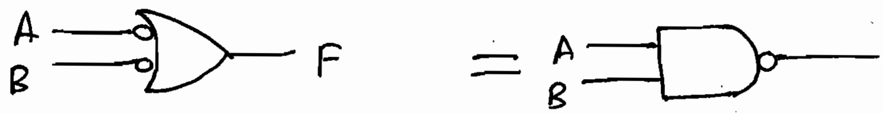
E.g. = $A \oplus B \oplus C = A \odot B \odot C$.

→ $\boxed{\text{Ex-NOR} = \overline{\text{EX-OR}}}$ if no. of Input Variables are even.

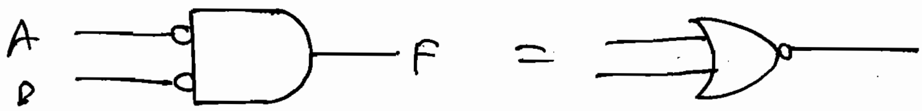
E.g. = $A \odot B \odot C \odot D = \overline{A \oplus B \oplus C \oplus D}$

7) Bubbled gates (Negative gates)

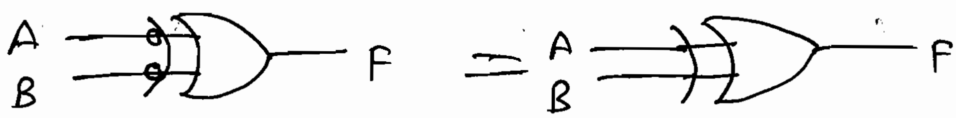
a) Bubbled OR gate = NAND gate



b) Bubbled AND gate = NOR gate

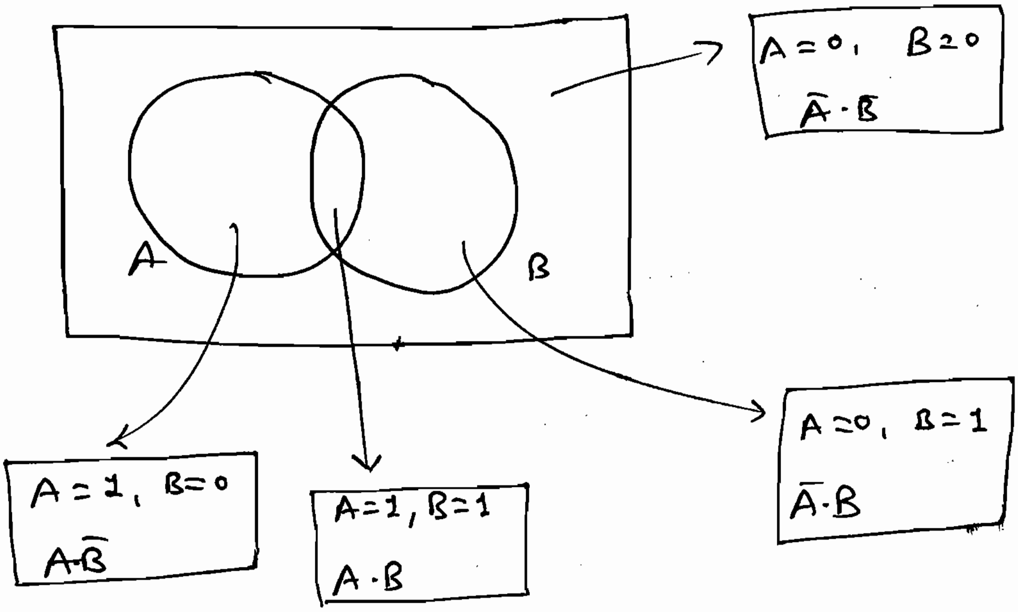


c) Bubbled Ex-OR = Ex-OR gate

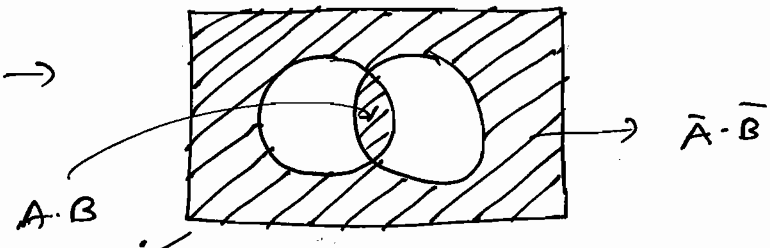


⑧ Venn diagrams:

→ **NOTE:**



→

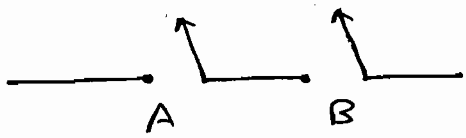


$$F = \bar{A} \cdot \bar{B} + A \cdot B$$

$F = A \odot B$ → Ex-NOR gate

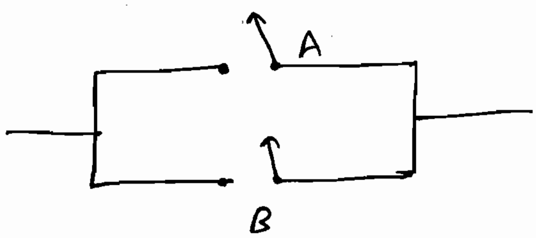
⑨

ⓐ



⇒ $F = A \cdot B.$ AND

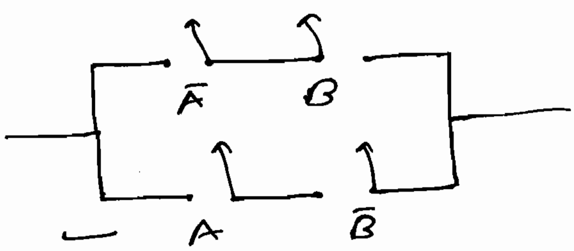
ⓑ



⇒ $F = A + B.$ OR

*

ⓒ

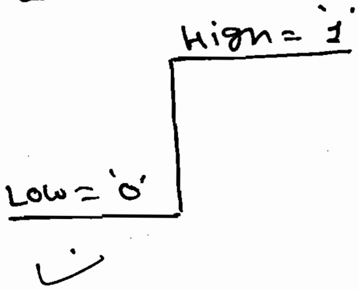


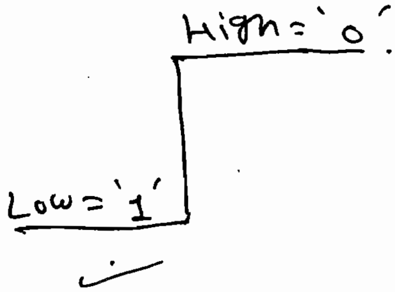
⇒ $F = \bar{A}B + A\bar{B}$

Ex-OR

F = Ex-OR gate.

(10) Positive Logic — Negative Logic

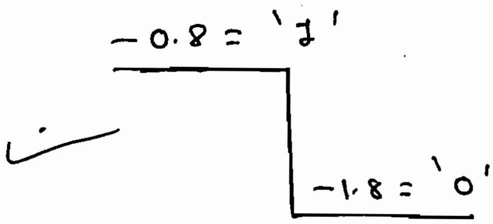




e.g. ECL Logic

$\Rightarrow$ '0' = −1.8

'1' = −0.8



→ Positive Logic.

NOTE: In negative logic, more negative value treated as logic '1' state. e.g. −1.8 → Logic '1' , −0.8 → as logic '0'.

Ex-1

(a) If $A \cdot B = 0$; EX-OR gate behaves ______ logic gates?

$\rightarrow$ $A \cdot B = 0$ $\qquad$ $A \cdot B = 1$

$\Rightarrow$ $A = 0,\ B = 1$ $\qquad$ $A = 1,\ B = 1$

$A = 1,\ B = 0$

$A = 0,\ B = 0$

$$\therefore\ A \oplus B = A\bar{B} + \bar{A}B = \overline{AB + \bar{A}\bar{B}}$$

$$= \overline{0 + \bar{A}\bar{B}}$$

$$= A + B$$

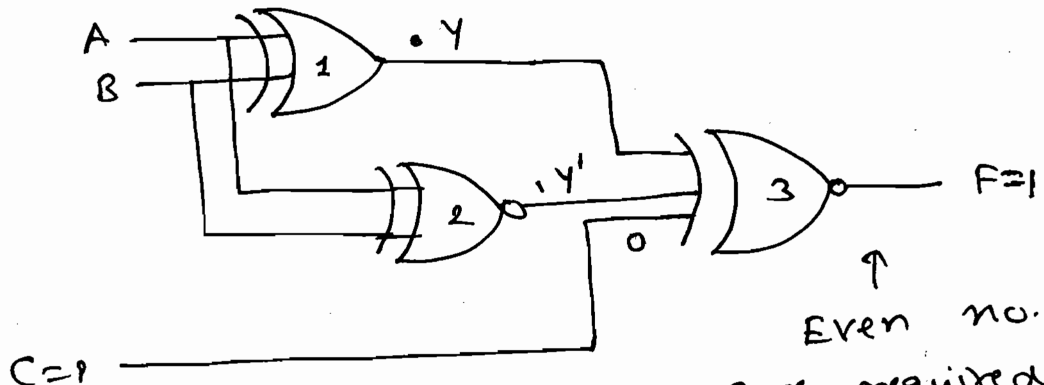
$$= \text{OR gate}$$

Imp NOTE:

In TTL Logic family open input is accepted by the logic gate as '1'

see: Page-133, Q=30 of IES '13 Book ACE Academy.

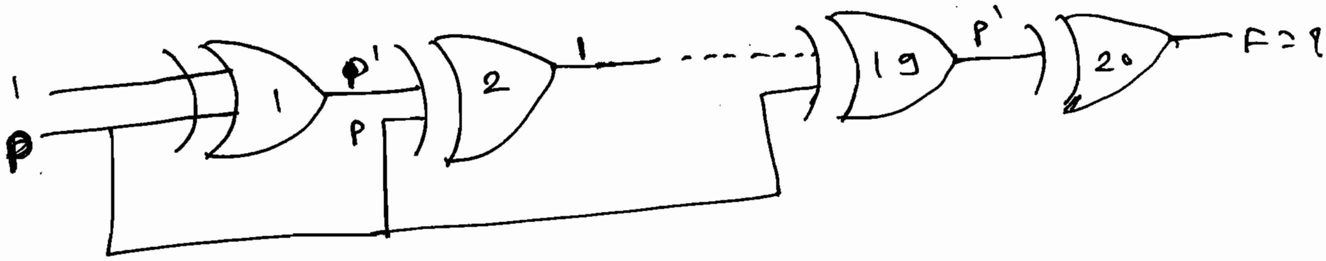
(b) Find the Value of C = ?



↑ Even no. of zeros are required at gate-3 to get F=1.

C = 0.

(c) Find output F = ?



~~P=~~ P' F = 1.

→ Output at even gate is 1 and at odd gate is P'

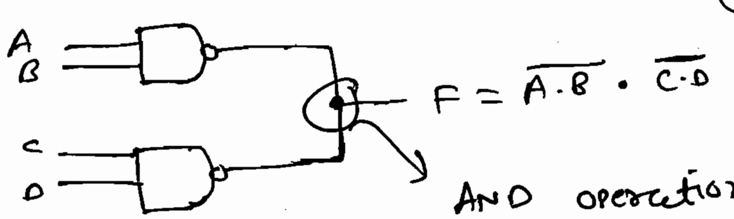
→ After even no. gates → O/P ⇒ 1 } → if IP = 1.

After 20 gates → O/P = 1.

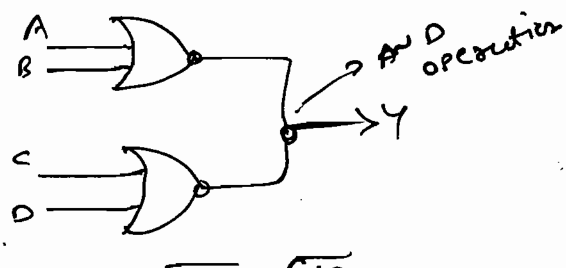
if initial condition is reverse that 1 become 0 the it act as a <u>buffer.</u>

NOTE: Open collector TTL will Provide wired-AND operation.

for eg.



(2)



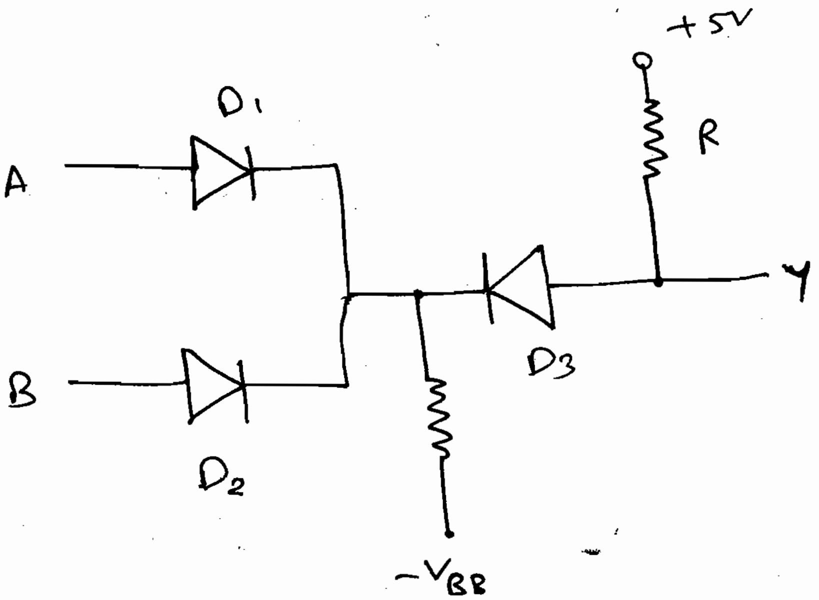
$Y = \overline{A+B} \cdot \overline{C+D}$

Ex - 2 Determine the Logic represented by the following Ckt.

(a) in +ve Logic.

(b) in -ve logic.

→



A	B	Y	
0	0	0 V	D_1, D_2, D_3 are ON.
+0	+5	+5 V	D_2 ON, D_1 & D_3 are OFF
+5	0	+5 V	D_1 ON, D_2 & D_3 are OFF
+5	+5	+5 V	D_1, D_2 ON, D_3 OFF.

$Y = A + B$ = OR gate.

(a) Positive logic.

+5 V → '1'

0 V → '0'

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

⇒ OR gate.

(b) Negative Logic

+5 → '0'
0 → '1'

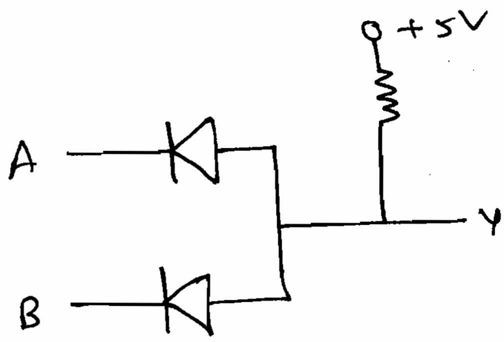
A	B	F
1	1	1
1	0	0
0	1	0
0	0	0

⇒ AND gate.

NOTE: (i) $\boxed{\text{+ve logic OR gate = -ve logic AND gate}}$

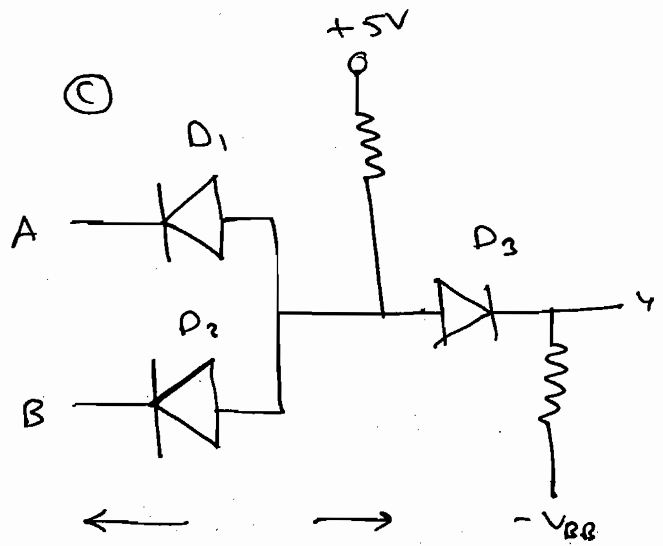
+ve logic		-ve logic
AND	→	OR
OR	→	AND
NAND	→	NOR
NOR	→	NAND
EX-OR	→	EX-NOR
EX-NOR	→	EX-OR.

(b)



← AND gate

(c)



← → AND gate

NOTE:

a) Diode pointing $\boxed{\text{outwards}}$ ⟹ $\boxed{\text{AND gate}}$

b) Diodes pointing $\boxed{\text{Inwards}}$ ⟹ $\boxed{\text{OR gate}}$

☆ Number Systems: (Positional weighted Number system)

		Base / Radix
1)	Decimal	10 (0, 1, 2, ..., 9)
2)	Binary	2 (0, 1).
3)	Hexadecimal	16 (0, 1, ..., 8, 9, A, B, C, D, E, F).
4)	octal	8 (0, 1, ... 6, 7).

Ex-1 → $\boxed{\text{Digit '6'} \Rightarrow \text{Base} \geq 7.}$

$b \cancel{=} 6$ ⟹ 0, 1, 2, 3, 4, 5, 6

∴ $b=7$ ⟹ 0, 1, ..., 6.

∴ $b=8$ ⟹ 0, 1, ..., 6, 7.

→ Digit '9' ⟹ Base $\geq$ 10.

→ Digit 'E' ⟹ Base $\geq$ 15.

NOTE:

Diode pointing	+ve Logic	-ve Logic
⟶ ⟵	OR	AND
⟵ ⟶	AND	OR

① $1101101.10101_2 = X_{16}$.

$$\rightarrow \quad \underline{110}\ \underline{1101}\ .\ \underline{1010}\ \underline{1}$$

$$\underline{0110} \downarrow 6 \qquad 13 \downarrow D \qquad 10 \downarrow A \qquad 1000 \downarrow 8$$

Ans: $(6D.A8)_{16}$.

Ex-2: How many bits are required to represent 6728_{10} in binary.

Ans: $2^n > 6728_{10}$

$n = 13$ bits. $\quad 2^{12} = 4096$

$2^{13} = 8192$

Ex-3: $6728_{10} \longrightarrow X_2 =$

$6728_{10} \searrow X_{16} \nearrow X_2$

16	6728	
16	420	8
16	26	4
	1	10 (A)
		1

$(1A48)_{16} = (6728)_{10}$

Now, 1 A 4 8

$\downarrow\ \downarrow\ \downarrow\ \downarrow$

0001 • 1010 0100 1000

$(6728)_{10} = (0001101001001000)_2$

Ex-3 How many bits are required to represent a 32 digit decimal no.?

Ans:

$$2^n > 10^{32}$$

$$n \ln 2 > 32 \ln 10$$

$$\therefore n > 32\left(\frac{\ln 10}{\ln 2}\right).$$

$$n > 106.30 \Rightarrow \boxed{n = 107}$$

Ex-4 Determine the base of the following relations.

(a) $24 + 17 = 40 \Rightarrow$ max digit $= 7$ hence base ≥ 8.

Let, base = 'b'

$$\therefore (2b^1 + 4b^0) + (1b^1 + 7b^0) = (4b^1 + 0).$$

$$\therefore 2b + 4 + b + 7 = 4b.$$

$$\therefore \boxed{b = 11}$$

Note:

$$36_8 \xrightarrow{3\times 8^1 + 6\times 8^0} X_{10}$$

$$AF_{16} \xrightarrow{10\times 16^1 + 15\times 16^0} X_{10}$$

$$24_b \xrightarrow{2\times b^1 + 4\times b^0} X_{10}$$

(b) $\sqrt{41} = 5$.

→ Let, Base = b.

$$\sqrt{4 \times b^1 + 1 \times b^0} = 5 \times b^0$$

$\therefore \quad 4b + 1 = 25$

$4b = 24$

$\therefore \boxed{\cancel{b \neq 4}}, \quad \boxed{b = 6}$

(c) Roots of $x^2 - 11x + 22 = 0$ are 3 and 6

$b = ?$. Max digit = 6, Base ≥ 7.

Ans:

$$\therefore (x-3)(x-6) = 0$$

$$\therefore x^2 - 9x + 18$$

let, roots are $x_1 = 3$, $x_2 = 6$.

$\therefore \quad x_1 + x_2 = -b/a$.

$3_{base} + 6_{base} = -\frac{(-11)}{1} = 11_{base}$

$\therefore \quad (3 \times base^0) + (6 \times base^0) = (base + 1)$

$\therefore \quad 3 + 6 = base + 1$

$\boxed{base = 8}$

(OR)

$x_1 \cdot x_2 = c/a$.

$\therefore \quad 3_{b_1} \times 6_{b_1} = \frac{22}{1} = 22_{b_1}$

$\therefore \quad 3 \times 6 = 2b_1 + 2$.

$\therefore \quad 18 = 2b_1 + 2$

$2b_1 = 16$

$\boxed{b_1 = 8}$

Ex 5 What is the min decimal value of

$11C_x = ?$

max digit C, so base ≥ 13.

Ans: $(x^2 + x + 12x^0)_{10}$ Base ≥ 13.

$= (x^2 + x + 12)_{10}$

Min decimal occures when base is minimum

i.e. $b = 13$.

$\therefore = 13^2 + 13 + 12$

$= 194_{10}$.

Ex 6

$$\begin{array}{r} 1.2_4 \\ + \; 2.3_4 \\ \hline (10.1)_4 \end{array}$$

1. $(2 \times 4^{-1}) = (1.5)_{10}$

2. $3/4 = (2.75)_{10}$

$\overline{4.25}$

↓

$0.25 \times 4 = 1.0$ ↑ $= 1$

$10.(\frac{2}{4} + \frac{5}{16})$

10.1.

Ex 7

$$\begin{array}{r} A B \\ + \; 3 D \\ \hline 200 \end{array}$$

$B = 11$

$D = 13$

$\overline{24}$

$A = 10$

$+ 3$

$+ 3$

$\overline{16}$

$8 \,|\, 24$ 3   0

$8 \,|\, 16$ 2   0

Ex 8 A

b)

$$\begin{array}{r} \overset{1}{A}\ B_H \\ +\ 3\ D_H \\ \hline (E\,8)_{16} \end{array}$$

$B = 11$

$$\begin{array}{r} D = \underline{13} \\ (24)_{10} \end{array}$$

$$\begin{array}{c|l} 16 & 24 \quad 8 \\ \hline & 1 \end{array}$$

$(18)_{16}$

$$\begin{array}{r} A = 10 \\ +\ 1 \\ +\ 3 \\ \hline 14 \end{array}$$

Ex - 8

$$\begin{array}{r} b=1 \quad \overset{11}{\cancel{1}}\ 1D_H \\ 2\ D_H \\ -\ A\ E_H \\ \hline -\ 7\ F_H \end{array}$$

$D = 13$

$E = 14.$

$11_{16} \rightarrow 17_{10}$

$\therefore 17 - 10 = 7$

$$\begin{array}{rcl} 1D_H & = & 29_{10} \\ -\ E_H & = & -14_{10} \\ \hline F_H & \leftarrow & 15_{10} \end{array}$$

☆ Complemetary Number Representation:

→ $A - B = A + (-B)$

$A - B = A + (\text{complement of } +B).$

→ Base = 'r' system

→ $\boxed{(r-1)'s \quad \text{Complement} \Rightarrow r^n - r^{-m} - N}$

→ $\boxed{r's \quad \text{Complement} \Rightarrow r^n - N.}$

∴ N = Given Number.

n = no. of digits in Integer part of N.

m = no. of digits in Fractional part of N.

E.g.: Find 9's Complement of 835.27_{10} = ?

→ $r = 10, \quad N = 835.27$ ← method-1

∴ $n = 3, \quad m = 2.$

∴ $10^3 - 10^{-2} - 835.27$

$= 1000 - 0.01 - 835.27$

$= 164.72_{10}$

NOTE: method:-2

$$\boxed{\begin{array}{r} 999.99 \\ -835.27 \\ \hline (164.72)_{10} \end{array}}$$

Ex-2 10's Complement $(352)_{11} = ?$

Ans: $(352)_{11}$. $r = 11$.

we have to find $(r-1)$'s Comp.

$$\therefore \quad \begin{array}{r} 10 \\ -3 \\ \hline 7 \end{array} \quad \begin{array}{r} 10 \\ -5 \\ \hline 5 \end{array} \quad \begin{array}{r} 10 \\ -2 \\ \hline 8 \end{array}$$

Ans: $= (758)_{11}$.

Ex-3 find 2's comp. of $x = 101101000$

Ans:

Complement 10110 | 1000 seek bit

copy

2's com: 01001 1000

Ex-4 $0110_2 - 0001_2$

$6_{10} - 1_{10}$

Ans: 0110_2 + (Complement of 0001).

→ **By 1's complement**

$$\begin{array}{rl} & 0110_2 = 6_{10} \\ + & 1110 \leftarrow \text{1's Complement of } 0001_2 \\ \hline \text{EAC } 1 & 0100 \\ & +1 \\ \hline & 0101 = 5_{10} \end{array}$$

→ **By 2's Complement.**

$$\begin{array}{rl} & 0110_2 \\ \text{EAC } + & 1111 \leftarrow \text{2's Comp. of } 0001_2 \\ \hline \cancel{1} & 0101 = 5_{10} \end{array}$$

EAC is discarded.

EAC = End Around Carry.

D: 24/6/2013

NOTE:

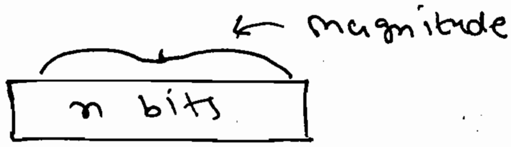
→ EAC doesn't occur when the result is −ve. i.e. Large value is subtracted from a small value.

(*)

1's Comp. term	2's Comp. term
→ +0 = 0000	→ +0 = 0000
−0 = 1's comp. of +0	→ −0 = 2's comp. +0
= 1's comp of 0000	= 2's comp. 0000
→ −0 = 1111	→ −0 = 0000.

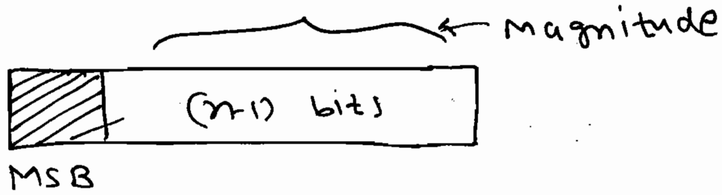
☆ Binary Numbers:

→ 1) Unsigned numbers. ⇒



2) Signed numbers. ⇒

⇓



sign bit

0 → +ve

1 → −ve

⇒ (a) signed magnitude form.

(b) 1's complement form.

(c) 2's complement form.

→ The Positive no. representation in all the above three forms is similar.

⇒ (a) sign magnitude form.

$+5_{10} = 0101$

$-5_{10} = 1101.$

(b) 1's Complement form.

$+5_{10} = 0101$

$-5_{10} =$ 1's comp. of +5

$-5_{10} = 1010$

(c) 2's Complement form.

$+5_{10} = 0101$

$-5_{10} =$ 2's comp. of +5.

$-5_{10} = 1011.$

Ex-1 Represent the following no. in 2's complement form:

a) -17_{10} b) -83.375_{10}.

Ans: a) $+17_{10} = \cancel{0}\cancel{0}010001$ ← for +ve sign.

$-17_{10} =$ 2's comp. of +17.

$= 101111$ -- 128 64 32 16 8 4 2 1.

0.5 0.25, 0.125 ...

b) -83.375_{10}

→ $83.375 =$ ~~0110001~~ 01010011. 011

$\rightarrow -83.375_{10}$ = 2's Comp. of $+83.375_{10}$

= 2's Comp. 01010011.011

= $(10101100.101)_2$

Ex-2

Determine the decimal values represented by the following signed no.s.

1) ~~XXØX is Ø101~~

1) sign mag. no. 1101 is $\underline{-5_{10}}$

$-5 = \underbrace{1}_{-}\underbrace{101}_{5} = \underbrace{0}_{+}\underbrace{101}_{} = +5$

2) (a) 2's comp 0110 is $\underline{+14}$

$\underbrace{0}_{+ve}\underbrace{1110}_{+14}$

~~XXXX~~

b) 1's comp no 01110 is $\underline{+14}$

$\underbrace{0}_{+ve}1110$

3) 2's comp no, 11001 is ⇒ ______ ?

$\underbrace{1}_{-ve}1001$ & 2's comp.

So, 11001

-sign 2's

→ -0 0111

= -7

-2^4	2^3	2^2	2^1	2^0
1	1	0	0	1

= -16 + 8 + 1

= -7.

4) 2's Comp. number 1000 is ⇒ $\underline{-8}$?

→ 1000

$= -8 + 0 + 0 + 0$

$= -8.$

$\underbrace{1000}$ ↓ 2's comp

1000

$= -8.$

5) 1's Comp. no 10010 is ____ ?

Ans:

$\underbrace{10010}$ 1's comp ↓

$- \cancel{0}1101$

$= -13$

$-(2^4-1)\ 2^3\ 2^2\ 2^1\ 2^0$

1 0 0 1 0

↓

$-15 + 0 + 0 + 2$

$= -13.$

GATE

(1) What is the equivalent 2's Comp. representation of a 2's Comp. no. 1101 is ____ ?

→ 2's Comp. no. → ~~~~ 1101. ⇒ -3

$= +3.$

(A) 101101 ⟶ -19_{10}

× (B) 001101 ⟶ $+13_{10}$

✓ (C) 111101 ⟶ -3_{10}

× (D) 011101 ⟶ $+29_{10}$

111101 2's

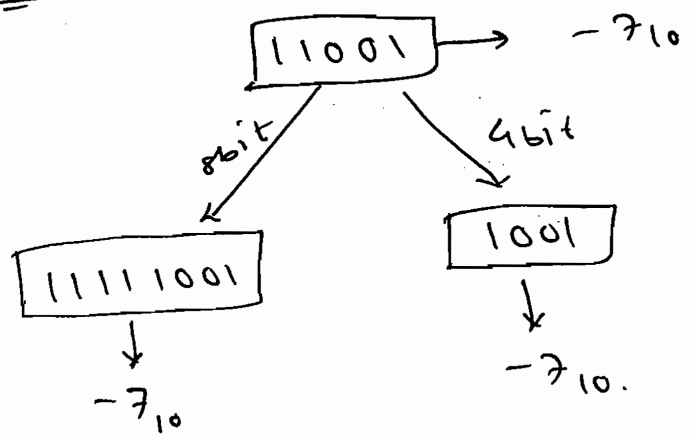
$= 0000\cancel{0}1$

$= -3.$

* Sign bit Extension:

→ In 1's & 2's Complement form the sign bit can be extended towards left any no. of times without changing its value.

E.g. Consider 2's comp. no.



Ex-1 A 2's Comp. no. "$x_4 x_3 x_2 x_1$" is copied into 6-bit register which of the following indicate the value of the register.

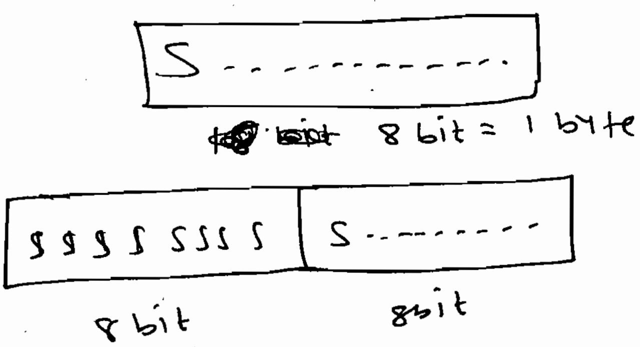
(a) $1 1 x_4 x_3 x_2 x_1$.

(b) $x_4 x_4 x_4 x_3 x_2 x_1$.

(c) $0 0 x_4 x_3 x_2 x_1$.

(d) None.

* Convert Byte to word (CBW).



2) Convert Word to Double word. => (CWD).

Word ↓ 16 bit

Double word ↓ 32 bit

Ex. 2

⇒ A register contain a 2's comp. no. 10100. if it is devided by 2 find the value of the reg.

Ans:

10100

$= -16 + 4 = -12.$

$\therefore \frac{-12}{2} = -6.$

~~2's comp -6~~

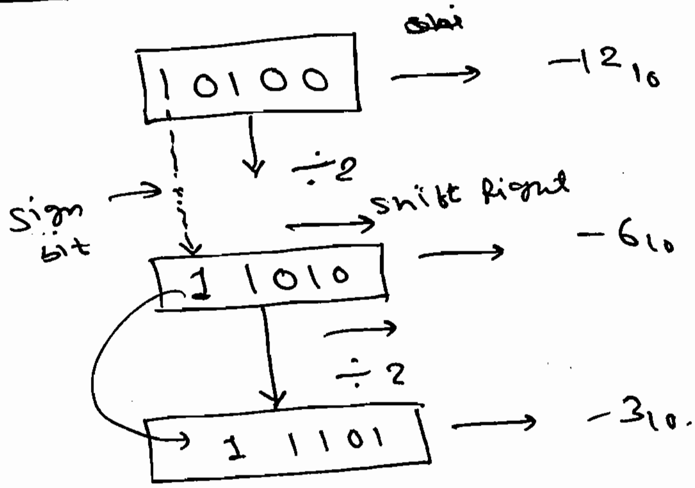
$\therefore$

$-6 =$ 2's comp. +6

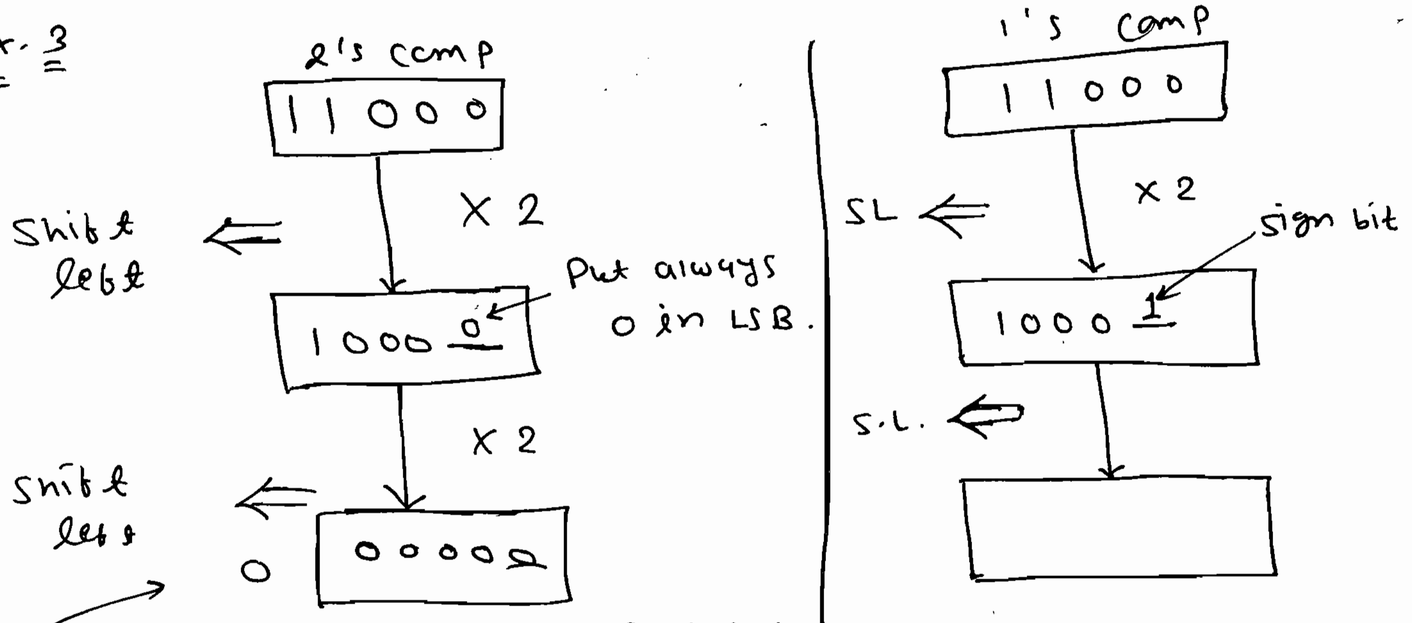
= 2's comp. 00110

= 11010

→ method-2:



Ex. 3

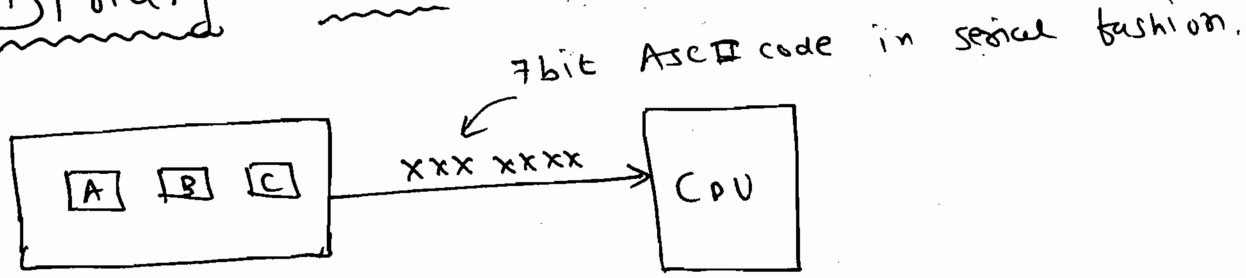


overflow (Range of 2's comp. nos. using 5 bit is exceeded.

	Range of number represents using 'n' bits.
1's comp. form, sign mag. form	$+(2^{n-1}-1)$ to $-(2^{n-1}-1)$. e.g. $n=5 \rightarrow +15_{10}$ to -15_{10}
2's comp form	$+(2^{n-1}-1)$ to -2^{n-1}. e.g. $n=5 \Rightarrow +15_{10}$ to -16_{10}

Note: Range of sign mag. form = Range of 1's comp. form.

☆ Binary Codes:



zero → '0' $\Rightarrow 30_H = \cancel{0}\boxed{0110000_2}$

'A' $\Rightarrow 41_H = \cancel{0}\boxed{1000001_2}$

'a' $\Rightarrow 61_H = \cancel{0}\boxed{1100001_2}$

(1) Alphanumberic Codes.

(2) Numeric Codes.

→

1) Alphanumeric codes
- → ASCII Code (7 bits; $2^7 = 128$ Alphanumericals)
- → EBCDIC code (8 bit = $2^8 = 256$ Alphanumericals)

→ used in IBM Computers.

2) Numeric Codes:

→ BCD (Binary coded Decimals) codes.

- Weighted
  - Positively
    - → 8 4 2 1 code (Natural BCD code)
    - → 5 4 2 1
    - → 3 3 2 1
    - → 2 4 2 1
  - negatively
    - → 8 4 −2 −1 code.
    - → 6 3 1 −1 Code.
- Non-weighted codes
  - Gray code (Unit distance code)
  - Excess −3

(Self Complementry Codes
sum of weights = 9)

Decimal Digit	8 4 2 1	Excess - 3
0	0 0 0 0	0 0 1 1
1	0 0 0 1	0 1 0 0
2	0 0 1 0	0 1 0 1
3	0 0 1 1	0 1 1 0
4	0 1 0 0	0 1 1 1
5	0 1 0 1	1 0 0 0
6	0 1 1 0	1 0 0 1
7	0 1 1 1	1 0 1 0
8	1 0 0 0	1 0 1 1
9	1 0 0 1	1 1 0 0
Invalid BCD 10	1 0 1 0	
⋮	: : : :	
16	1 0 1 1	

(1) Self Complementary.

	8421	EX-3	
3_{10} =	0011	0110	→ 3_{10}
9's Comp ↓		↓ 1's Comp.	
6_{10} =	~~0000~~ 0110	1001	→ 6_{10}

(2) Sequential Codes:

	8421	Ex-3	
3_{10} →	0011	0110	← 3_{10}
	+ 1	+ 1	
4_{10} →	0100	0111	← 4_{10}
	+ 1	+ 1	
5_{10} →	0101	1000	← 5_{10}
	✓	✓	

→ The advantage of Ex-3 code is it is both sequential and self complementary.

→ During BCD addition output is invalid

(1) If the result is greater than 9

(2) If carry occurs during BCD addition.

E.g.

68_{10} → 0110 1000 (carry = 1 → Invalid)

$+58_{10}$ → + 0101 1000

126_{10} → 1100 0000 (>9 → invalid)

+ 0110 0110

1 0010 0110

Ans → 126_{10}

(2) Find the no. of BCD correction = ?

@ 174_{10} → 0001 0111 0100

$+826_{10}$ → + 1000 0010 0110

1000 → 1001 1001 1010 → >9 ×

+ 0110

1001 1010 0000

+ 0110 0000

1010 0000 0000

+ 0110 0000 0000

1 0000 0000 0000

1 0 0 0

$= 1000_{10}$

So, 3 correction

* Excess-3 Addition

① If carry occur → Add 3_{10}

② If carry doesn't occur → Subtract 3_{10}.

e.g.

$$
\begin{array}{rcl}
38_{10} & \rightarrow & 0110\ 1011 \\
+\ 46_{10} & \rightarrow & 0111\ \cancel{0110}\ 1001 \\
\hline
 & & 1110\ 0100 \\
 & & -\ 0011\ \ +\ 0011 \\
\hline
 & & 1011\ \ 0111
\end{array}
$$

1011 ↓ 11 − 3 ↓ 8

0111 ↓ 7 − 3 ↓ 4

⇒ 84_{10}

* Grey Code (Reflected code) (unit distance code) (cyclic code).

1-bit Gray code $\xRightarrow{\text{Reflective property}}$ 2-bit gray code $\xRightarrow{RP}$ 3-bit gray code

1-bit	2-bit	3-bit	
0	00	000	(0)
1	01	001	(1)
	11	011	(2)
	10	010	(3)
		110	(4)
		111	(5)
		101	(6)
		100	(7)

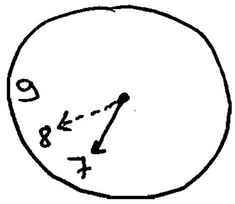
dist. unit = 1

dist=1 (Cyclic Code)

→ Use of grey Code:

① K-map
② Shaft encoders
③ Error correction and detection
④ Genetic algorithm.

→ Shaft encoders



8421	Gray code
7 → 0111	0100
8 → 1000	1100

More chance to get error (or) all 4 bit should change

Less chance to get error as it is unit distance code.

☆ Code Conversion

① Binary to Gray.

$B_4 \oplus B_3 \oplus B_2 \oplus B_1$

Binary:	1	1	0	1
↓	↓	↓	↓	↓
Gray:	1	0	1	1
	G_4	G_3	G_2	G_1

$G_4 = B_4$

$G_3 = B_4 \oplus B_3$

$G_2 = B_3 \oplus B_2$

$G_1 = B_2 \oplus B_1$

→ Ex-OR → Modulo - 2 Addition

0 + 0 = 0

0 + 1 = 1

1 + 0 = 1

1 + 1 = 0.

(2) Gray to Binary.

	G_4	G_3	G_2	G_1
Gray :	1	0	1	1
	↓	⊗ ↓	⊗ ↓	⊗ ↓
Binary :	1	1	0	1
	B_4	B_3	B_2	B_1

$B_4 = G_4$

$B_3 = B_4 \oplus G_3$

$B_2 = B_3 \oplus G_2$

$B_1 = B_2 \oplus G_1$

Ex-1 Represent $(743)_8$ in Gray code.

Ans:

$(743)_8$

↓

$(\underset{7}{\underline{111}}\ \underset{4}{\underline{100}}\ \underset{3}{\underline{011}})_2$

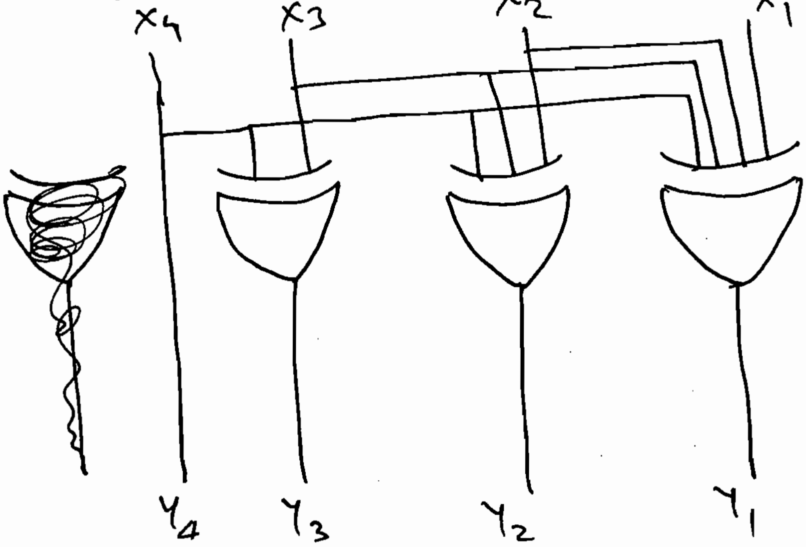
Now B: 1 1 1 1 0 0 0 1 1

↓

G: $(100010010)_2$

$\therefore (743)_8 =$ 100010010 ← Gray code.

Ex-2 Identify the following code converter.



$Y_4 = X_4$

$Y_3 = X_4 \oplus X_3$

$Y_2 = X_4 \oplus X_3 \oplus X_2$

$Y_3 = X_4 \oplus X_3 \oplus X_2 \oplus X_1.$

& G → B

Ex-3 $b_4 b_3 b_2 b_1$ is a 4-Bit binary no. What is the f^n of the following eq^ns.

(1) $b_{11}' = b_1$.

$b_{22}' = b_1 \oplus b_2$.

$b_{33}' = (b_1 + b_2) \oplus b_3$.

$b_{44}' = (b_1 + b_2 + b_3) \oplus b_4$.

b_1	b_2	b_3	b_4
0	1	0	1
0	1	1	0
0	1	1	0
1	0	1	0
0	1	1	0

So, Ans is 2's complement.

$(b_1+b_2+b_3) \oplus$ $\qquad$ $\oplus (b_1+b_2)$

$b_4 \quad b_3 \quad b_2 \quad b_1$

$\downarrow \quad \downarrow \quad \downarrow \quad \downarrow$

$b_{44} \quad b_{33} \quad b_{22} \quad b_{11}$

* Hamming Code (Single Error Correcting Code):

→ $\boxed{2^k \geq m + k + 1}$

m = no. of message bits.

k = no. of parity bits.

E.g. $\boxed{m = 4}$ ⟹ No. of Parity bits $\boxed{k = 3}$

Let, m_1, m_2, m_3, m_4 $\qquad$ Let, P_1, P_2, P_3.

001	010	011	100	101	110	111
$2^0 = 1$	$2^1 = 2$	3	$2^2 = 4$	5	6	7
P_1	P_2	m_1	P_3	m_2	m_3	m_4

✓ Choose 'P_1' such that $1, 3, 5, 7 = P_1, m_1, m_2, m_4$ has odd parity

✓ Choose 'P_2' such that $2, 3, 6, 7 = P_2, m_1, m_3, m_4$ " " "

Choose 'P_3' such that $4, 5, 6, 7 = P_3, m_2, m_3, m_4$ has odd parity.

Ex-1 (7,4) Hamming Code with odd parity for the message 1001.

(a)

→ 7 = total no. of bits.

4 = no. of message bits.

1	2	3	4	5	6	7
P_1	P_2	1	P_3	0	0	1

Choose P_1 = P_1, 1, 0, 1 should have odd parity ⇒ ~~Cho~~

1, 3, 5, 7 ⇒ Choose $\boxed{P=1}$

Choose P_2 2, 3, 6, 7 = P_2 1 0 1 " " " "

$\boxed{P_2 = 1}$

Choose P_3 4, 5, 6, 7 = P_3 0 0 1 " " "

$\boxed{P_3 = 0}$.

Corrected code: 1 1 1 0 0 0 1

(b) (7,4) code hamming code is received as

1 1 1 0 1 0 1

Ans: 4, 5, 6, 7 ⇒ 0 1 0 1 ⇒ even parity ⇒ $C_3 = 1$
(in error)

2, 3, 6, 7 ⇒ 1, 1, 0, 1 ⇒ odd parity ⇒ $C_2 = 0$

1, 3, 5, 7 ⇒ 1, 1, 1, 1 ⇒ even parity ⇒ $C_1 = 1$.
(in error)

i.e. error occured at $C_3 C_2 C_1$ = 101 = 5th position.

Received Code = 1 1 1 0 1 0 1   transmitted code

Corrected code = 1 1 1 0 0 0 1   = 1001.

⇒ For Hamming Distance

a) For correcting 't' errors: $\boxed{\text{Hamming distance} \geq 2t+1.}$

b) For detecting 't' errors: $\boxed{\text{Hamming distance} \geq t+1.}$

☆ Boolean Algebra :

⇒ 'AND' Law — 'OR' Law

Identity element ⇒ '1' — '0'

$A \cdot 0 = 0$ — $A + \boxed{0} = A$

$A \cdot \boxed{1} = A$ — $A + 1 = 1.$

ⓒ Commutative Law:

Dual

$A + B = B + A.$

$A \cdot B = B \cdot A.$

NAND : ↑

Dual

$A \uparrow B = B \uparrow A$

$\therefore \overline{A \cdot B} = \overline{B \cdot A}$

$A \downarrow B = B \downarrow A$

⇒ Inhibition : (/)

⇒ $x / y = x \cdot \bar{y}$

$$\boxed{\begin{array}{ccc} A/B & \not\Rightarrow & B/A \\ \therefore A\cdot\bar{B} & \neq & B\cdot\bar{A} \end{array}}$$

→ 'Inhibition is not Commutative.'

(b) Associative Law :

Dual
$(A+B)+C = A+(B+C)$

$(A\cdot B)\cdot C = A\cdot(B\cdot C)$

$(\overset{1}{A}\uparrow\overset{1}{B})\uparrow\overset{0}{C} \neq \overset{1}{A}\uparrow(\overset{1}{B}\uparrow\overset{0}{C})$

$1\uparrow 0$

$0\uparrow 0$ $\qquad$ $1\uparrow 1$

$=1$ $\qquad$ $=0$

→ The NAND and NOR operation are Commutative but not Associative.

(c) Consensus Law:

$A\cdot B + \bar{A}\cdot C + B\cdot C = A\cdot B + \bar{A}\cdot C.$

$\therefore A\cdot B + \bar{A}C + B\cdot C(A+\bar{A})$

$= A\cdot B + \bar{A}C + ABC + \cancel{B\bar{A}} + \bar{A}BC$

$= \cancel{A\cdot B + \bar{A}C +}$

$= ABC(1+C) + \bar{A}C(1+C).$

$= AB + \bar{A}C$

$$\boxed{x\cdot y + \bar{y}\cdot z + WVXZ = x\cdot y + \bar{y}\cdot z}$$

Dual $(A+B)\cdot(\bar{A}+C)\cdot(B+C) = (A+B)\cdot(\bar{A}+C).$

(4) Distribution Law:

$\rightarrow$ $A\cdot(B+C) = AB + AC.$

Dual

$A + (B\cdot C) = (A+B)\cdot(A+C)$

i) $\bar{X} + X\bar{Y} = (\bar{X}+X)(\bar{X}+\bar{Y})$

$\therefore \boxed{\bar{X} + X\bar{Y} = \bar{X} + \bar{Y}}$

(ii) $AB + \overline{AB}C = (AB + \overline{AB})(AB+C).$

$= AB + C.$

(5) Transposition Law:

$\rightarrow$ $AB + \bar{A}C = (\bar{A}+B)(A+C).$

$= \bar{A}\cdot A + \bar{A}C + AB + BC.$

$= \bar{A}C + AB + BC.$

$= \bar{A}C + AB$ $(\because$ Consensus Law$).$

$=$ L.H.S

$\rightarrow$ $\boxed{x\cdot y + \bar{y}\cdot z = (x+\bar{y})(y+z).}$

⑥ De Morgan's Law:

→ ⓐ NOR gate = Bubbled AND gate

$$\boxed{\overline{A+B+C+\cdots} = \bar{A}\cdot\bar{B}\cdot\bar{C}\cdots}$$

→ ⓑ NAND gate = Bubbled OR gate.

$$\boxed{\overline{A\cdot B\cdot C\cdot D\cdots} = \bar{A}+\bar{B}+\bar{C}+\bar{D}+\cdots}$$

⑦ Shannon's Law:-

→ To find Complement of a fun 'F'.

(i)	Find the Dual of F i.e. 'F_D'.
(ii)	Complement all variables.

Ex-1 $F = AB + BC + CA$ then $\bar{F} = \bar{A}\cdot\bar{B} + \bar{B}\cdot\bar{C} + \bar{C}\cdot\bar{A}$

[T/F].

Ans:

$F = AB + BC + CA.$

$\therefore \cancel{\bar{F}} = \cancel{(\bar{A}+\bar{B})}\cdot\cancel{(\bar{B}+\bar{C})}\cdot\cancel{(\bar{C}+\bar{A})}.$

$\cancel{= (\bar{A}\bar{B} + \bar{A}\cdot\bar{C})\,(\bar{C}+\bar{A})}.$

$\cancel{= \bar{A}\bar{B} + \bar{A}\cdot\bar{C} + \bar{A}\bar{B}\bar{C} + \bar{A}\cdot\bar{C}}.$

$= \cancel{\bar{A}\bar{B} + \bar{A}\cdot\bar{C} +}$

i) $F_D = (A+B)\cdot(B+C)\cdot(C+A).$

(ii) $\bar{F} = (\bar{A}+\bar{B})(\bar{B}+\bar{C})(\bar{C}+\bar{A})$

$= (\bar{B} + \bar{A}\bar{C})\cdot(\bar{C}+\bar{A})$.

$= \bar{B}\bar{C} + \bar{A}\bar{B} + \bar{A}\bar{C} + \bar{A}\bar{C}$.

$\therefore \bar{F} = \bar{A}\bar{B} + \bar{B}\cdot\bar{C} + \bar{C}\cdot\bar{A}$.

** Ex-2 Simpify the following boolian expression to four literals.

① $F = \bar{A}C + \bar{C}D + \bar{B}C + AB$. $\longrightarrow$ 4 literals.

Note: Literal: variable (or) Complement of variables.

Eg: $F(A,B,C) \Rightarrow A, \bar{A}, B, \bar{B}, C, \bar{C}$.

$F = (\bar{A}+\bar{B})C + \bar{C}D + AB$.

$F = \overline{AB}\cdot C + \bar{C}D + AB$.

$\therefore F = AB + C + \bar{C}D$.

$\boxed{F = AB + C + D}$

Ex-3 Determine the number of two input NAND gates required to implement the following:

Ans: a) $F = (\bar{X}+\bar{Y})(W+Z)$.

$\therefore \bar{F} = \cancel{\overline{(\bar{X}+\bar{Y})\cdot(W+Z)}}$.

$= \cancel{XY}\cdot\cancel{\bar{W}\bar{Z}}$

$F = \overline{X\cdot Y}\,(W+Z)$.

$\therefore F = \overline{X\cdot Y}\,W + \overline{X\cdot Y}\cdot Z$

$= PW + PZ$

$P = \overline{X\cdot Y}$

$F = \overline{\overline{PW}\cdot\overline{PZ}}$

4 NAND gate req.

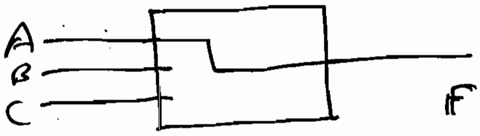
(b) $F = A + AB + ABC.$

$= A + AB\,(1+C).$

$= A + AB.$

$= A\,(1+B)$

$\therefore \boxed{F = A}$



'0' NAND gates.

$\Rightarrow$ 0 NAND gate is required.

(c) n - input AND gate = ?

NAND gates

$\rightarrow$ 2 i/p AND $\Rightarrow F = \overline{\overline{A\cdot B}}$ = (2)

3 i/p AND $\Rightarrow F = ABC$

$= \overline{\overline{\overline{\overline{AB}}\; C}}$ = (4)

4 i/p AND $\Rightarrow F = \overline{\overline{\overline{\overline{AB}}\;\overline{\overline{CD}}}}$ = (6).

=

⋮

n i/p AND $\Rightarrow$ ~~for~~ $(2n-2)$ no. of 2 i/p NAND gate req.

Ex-4 Implement EX-OR gate using minimum no. of (a) NAND gate (b) NOR gate.

Ans:

$\rightarrow A \oplus B = \bar{A}.B + A.\bar{B}$

$= \bar{A}\cdot B + A\cdot\bar{B} + \boxed{A\cdot\bar{A} + B\cdot\bar{B}}$

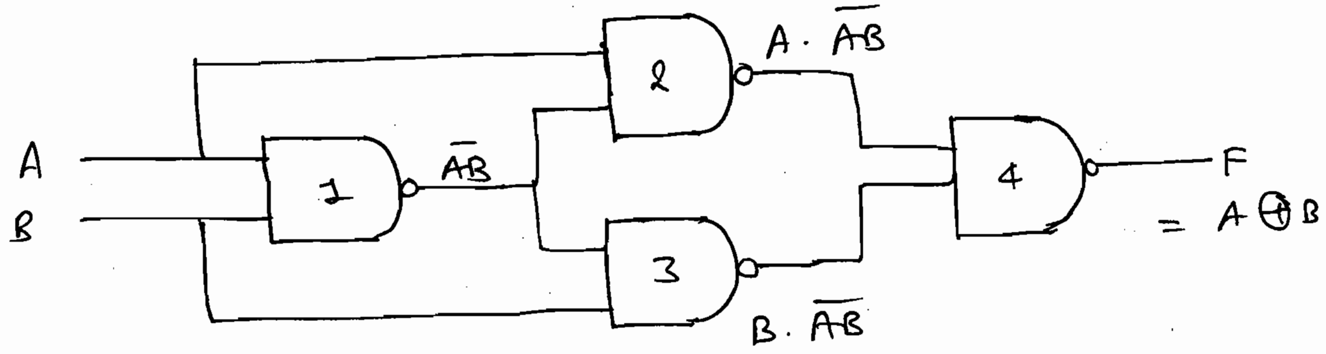
$= B\,(\bar{A} + \bar{B}) + A\,(\bar{A} + \bar{B}).$

$= A\cdot\overline{AB} + B\cdot\overline{AB}.$

$$\bar{\bar{F}} = \overline{\overline{A\cdot\overline{AB} + B\cdot\overline{AB}}}$$

$$\therefore \bar{F} = \overline{\overline{A\cdot\overline{AB}} \cdot \overline{B\cdot\overline{AB}}} .$$

→



(b) Using NOR gate:

$$F = A \oplus B$$

$$F = A(\bar{A}+\bar{B}) + B(\bar{A}+\bar{B}).$$

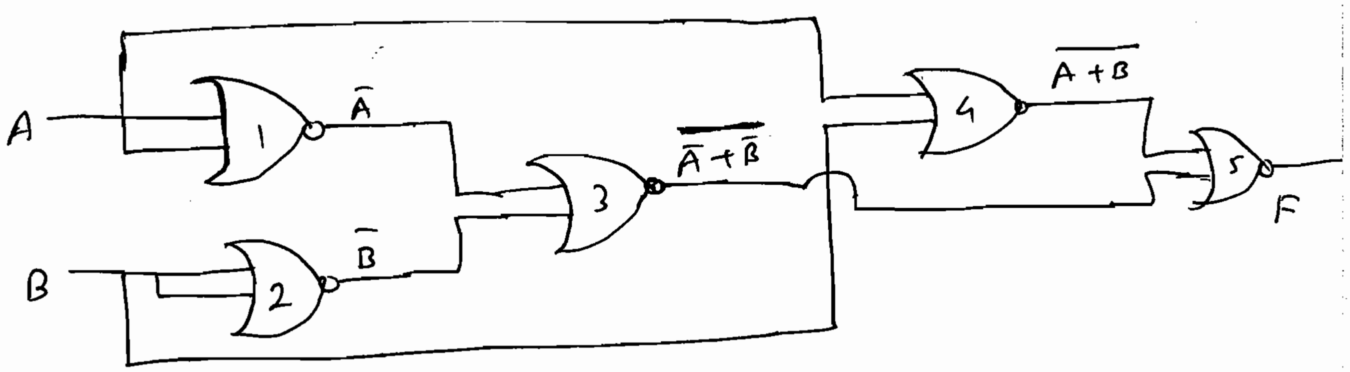
$$\therefore \cancel{F/F = \overline{\overline{A(\bar{A}+\bar{B}) + B(\bar{A}+\bar{B})}}}$$

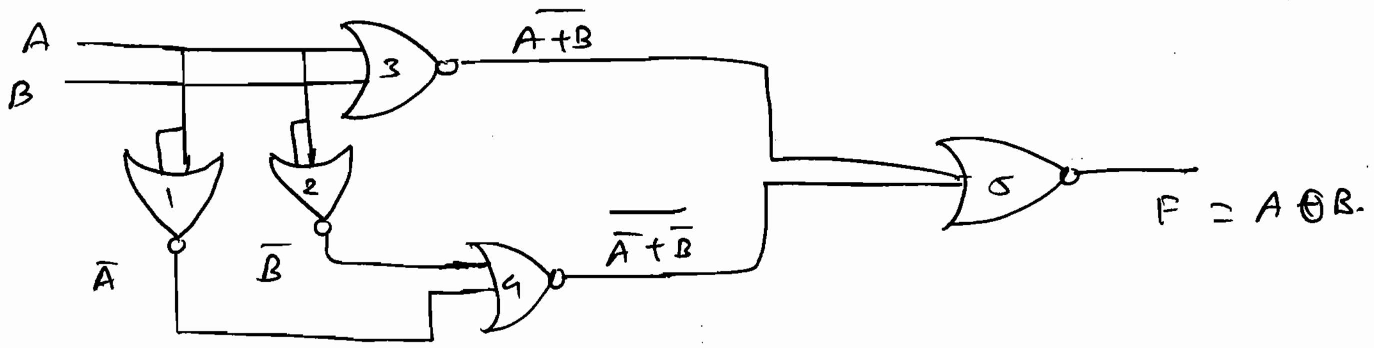
$$\bar{F} = \boxed{(\bar{A}+\bar{B})(A+B).}$$

$$= \overline{\overline{(\bar{A}+\bar{B})(A+B)}}$$

$$F = \overline{\overline{\bar{A}+\bar{B}} + \overline{A+B}}$$

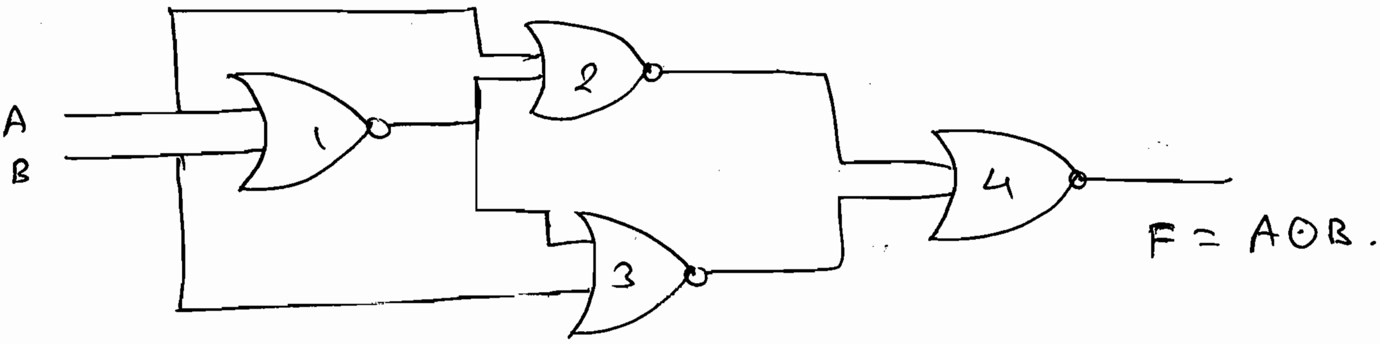
∴ 5 NOR gate.



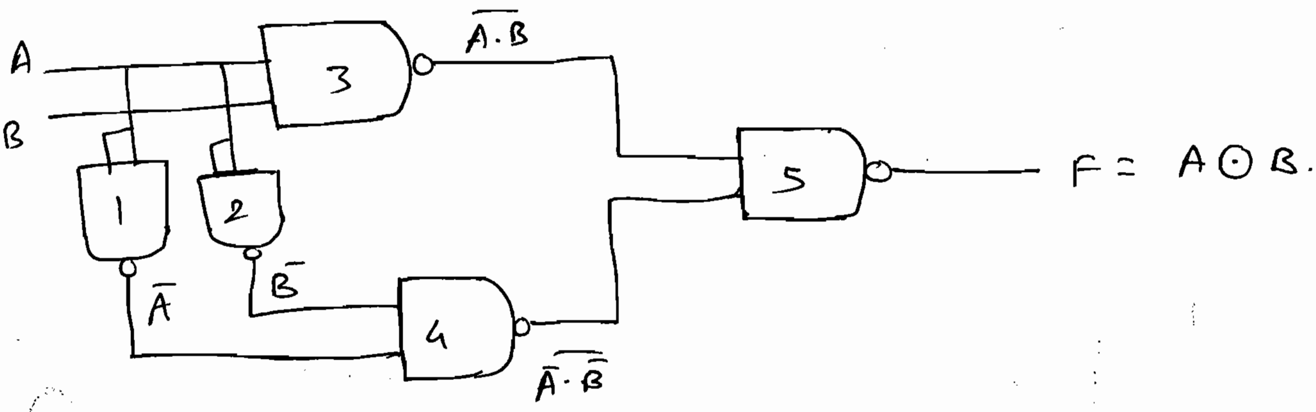


→ If complement of I/P variables are given i.e. $\bar{A}$, $\bar{B}$ are available then only 3 NOR gates are required to implement $A \oplus B$.

* X-NOR using min. no. of NOR gate:



* X-NOR using min. no. NAND gates.



* Minterms, Maxterms & Properties.

→ Minterms → Standard Product term.

→ maxterms → Standard Sum term.

⇒ Minterms → 8

m_0	$\overset{0}{\bar{A}} \cdot \overset{0}{\bar{B}} \cdot \overset{0}{\bar{C}}$
m_1	$\overset{0}{\bar{A}} \cdot \overset{0}{\bar{B}} \cdot \overset{1}{C}$
m_2	$\overset{0}{\bar{A}} \cdot \overset{1}{B} \cdot \overset{0}{\bar{C}}$
m_3	$\bar{A} \cdot B \cdot C$
m_4	$A \cdot \bar{B} \cdot \bar{C}$
m_5	$A \cdot \bar{B} \cdot C$
m_6	$\overset{1}{A} \cdot \overset{1}{B} \cdot \overset{0}{\bar{C}}$
m_7	$\overset{1}{A} \cdot \overset{1}{B} \cdot \overset{1}{C}$

$$\boxed{\begin{array}{l} Var = 1 \\ \overline{Var} = 0 \end{array}}$$

Maxterms → 8

M_7	$\overset{\phi}{\bar{A}} + \overset{\phi}{\bar{B}} + \overset{\phi}{\bar{C}}$
M_6	$\overset{1}{\bar{A}} + \overset{1}{\bar{B}} + \overset{0}{C}$
M_5	$\overset{1}{\bar{A}} + \overset{0}{B} + \overset{1}{\bar{C}}$
M_4	$\overset{1}{\bar{A}} + \overset{0}{B} + \overset{0}{C}$
M_3	$\overset{0}{A} + \overset{1}{\bar{B}} + \overset{1}{\bar{C}}$
M_2	$A + \bar{B} + C$
M_1	$A + B + \bar{C}$
M_0	$\overset{0}{A} + \overset{0}{B} + \overset{0}{C}$

$$\boxed{\begin{array}{l} Var = 0 \\ \overline{Var} = 1 \end{array}}$$

Ex-1 $F(A, B, C, D, E)$ and find $m_{23} = ?$, $M_{19} = ?$

Ans:

$$m_{23} = \overset{1}{A} \cdot \overset{0}{\bar{B}} \cdot \overset{1}{C} \overset{1}{D} \overset{1}{E}$$

$$M_{19} = \overset{1}{\bar{A}} + \overset{0}{B} + \overset{0}{C} + \overset{1}{\bar{D}} + \overset{1}{\bar{E}}.$$

* Properties:

① n-var Function ⇒ $\boxed{\begin{array}{l} 2^n \text{ minterms} \\ 2^n \text{ maxterms.} \end{array}}$

② $\boxed{M_j = \overline{m_i}}$ and vice versa.

③ $\boxed{m_i^D = M_{(2^n - 1 - i)}}$.

D = dual

e.g. $m_3 = \overset{0}{\bar{A}}\overset{1}{B}\overset{1}{C}$

$m_3^D = \overset{1}{\bar{A}} + \overset{0}{B} + \overset{0}{C}$

$= M_4$

$M_3^D = M_{2^3-1-3}$.

④ ⓐ Sum of all minterms = 1. $\boxed{\text{i.e } \sum_{i=0}^{2^n-1} m_i = 1.}$

ⓑ Product of all maxterms = 0 i.e. $\boxed{\prod_{j=0}^{2^n-1} M_j = 0.}$

BEL ☆☆

Ex-1 How many minterms are present at the o/p of 6 i/p Ex-OR gate: $2^{6-1} = 32$ no. of minterms.

Ans:

$A \oplus B = A \cdot \bar{B} + \bar{A} \cdot B$.

~~$A \oplus B \oplus C$~~ ~~=~~ ~~$A\bar{B}$~~

$A \oplus B = m_1 + m_2$ [2 out of 4 minterms].

$\therefore A \oplus B \oplus C = m_1 + m_2 + m_4 + m_7$ [4 out of 8 minterms]

$\therefore$ ~~$A \oplus B \oplus C$~~ n-input Ex-OR gate output Contains $= \frac{2^n}{2}$

$= 2^{(n-1)}$ no of ~~output~~ minterms.

NOTE: Same for X-NOR.

Ex-2 How many boolean f^ns can be formed as using n- boolean variables?

Ans: 'n' Boolean variable $\longrightarrow$ Boolean functions $= 2^{2^n}$.

$\Downarrow$

$x = 2^n$ minterms $\nearrow$ $2^x = 2^{2^n}$.

F (A, B)

	AB	F_0	F_1	F_2	F_3	....	F_{15}
$m_0 \leftarrow$	00	0	0	0	0		1
$m_1 \leftarrow$	01	0	0	0	0		1
$m_2 \leftarrow$	10	0	0	1	1		1
$m_3 \leftarrow$	11	0	1	0			1
		ϕ (Null)	AND	Inhibition (A|B)	Transfer		Identity.

* Forms of Boolean Functions:

1) a) Sum of Products (SOP) form. $\rightarrow$ DNF

b) Product of Sum (POS) form. $\rightarrow$ CNF

2) a) Canonical (or) Standard SOP form. (sum of minterms) $\rightarrow$ DCF

b) Canonical (or) Standard POS form. (Product of max terms) $\rightarrow$ CCF

- DNF = Disjunctive Normal form
- CNF = conjunctive normal form.
- DCF = Disjunctive canonical form
- CCF = Conjuctive Canonical form.

Ex-1 $F(A,B,C,D) = \overset{0}{\bar{A}} + \overset{010}{\bar{A}C\bar{D}} + \overset{01}{\bar{B}C}$ to canonical SOP form.
(sum of minterms form

Ans: ~~$F(A,B,C,D) = \bar{A}B + \bar{A}\bar{B} + \bar{A}BC\bar{D} + \bar{A}\bar{B}C\bar{D} +$~~

$F(A,B,C,D) = \bar{A} + \bar{A}C\bar{D} + \bar{B}C.$

$\bar{A}$	$\bar{A}C\bar{D}$	$\bar{B}C$
$\bar{A}$ — — —	$\bar{A}$ — $C\bar{D}$	~~0010~~
$m_0 \leftarrow$ 0 0 0 0	~~0 0 1 0~~	~~0 0 1 1~~
0 0 0 1	~~0 1 1 0~~	1 0 1 0 $\rightarrow m_{10}$
0 0 1 0		1 0 1 1 $\rightarrow m_{11}$
0 0 1 1		
0 1 0 0		
0 1 0 1		
0 1 1 0		
m_7 0 1 1 1		

$\therefore F(A,B,C,D) = \sum m(0,1,\ldots 7,10,11) \leftarrow$ Can SOP form.

$\therefore F(A,B,C,D) = \prod M(8,9,12,13,14,15) \leftarrow$ Can POS form.

Ex-2 Convert $F(A,B,C) = A\cdot B + \bar{A}\cdot C$ into Canonical POS form (product of maxterms)

Ans: ~~$F(A,B,C) =$~~ $A\cdot B + \bar{A}\cdot C$

~~m_0 000~~ ~~A 100 M_4~~

~~m_1 001~~ 101 M_5.

$\therefore$ SOP $\rightarrow$ Can SOP $\rightarrow$ Can POS

SOP $\rightarrow$ POS $\rightarrow$ Can POS.

$F = A\cdot B + \bar{A}\cdot C.$

$= \cancel{(A+B)\cdot(\bar{A}+C)}.$

$F = (A+C)\cdot(\bar{A}+B)$

$\boxed{F = \prod m(0,2,4,5).}$

	A	B	C
M_0	0	0	0
M_2	0	1	0

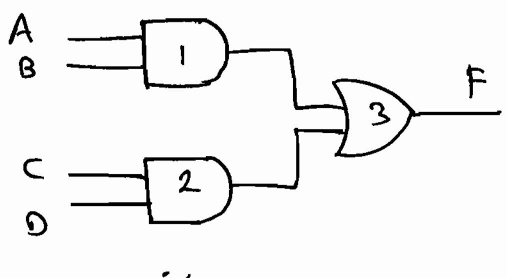
A	B	C	
ϕ	0	0	M_4
ϕ	0	1	M_5

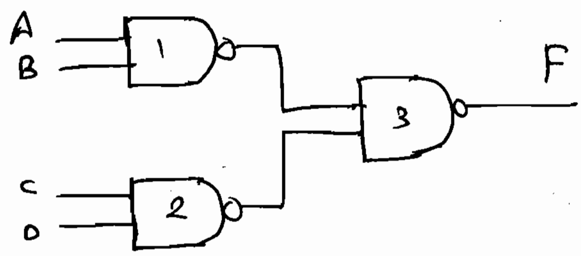
* Two Level Logic:

→ (a) **Sop form:**

$F = AB + CD$

AND - OR Logic ≡ **NAND - NAND Logic**

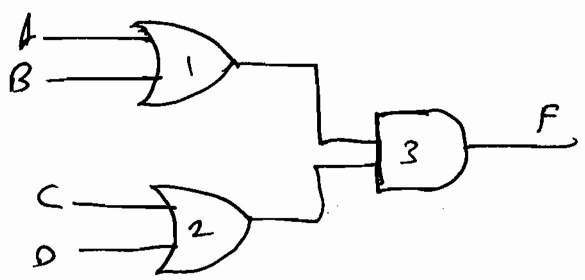


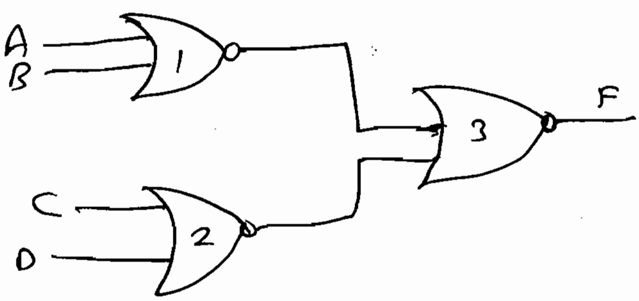


(2) **Pos - form:**

$F = (A+B)\cdot(C+D).$

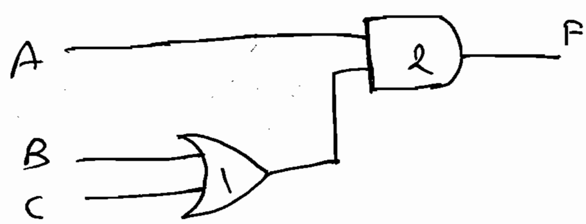
OR - AND Logic : ≡ **NOR - NOR Logic**





* Hybrid Logic.

$F = AB + AC = A \cdot (B+C)$



→ The advantages of two level logic is the propogation for all the input variables is same.

☆ Types of Two Level logic:

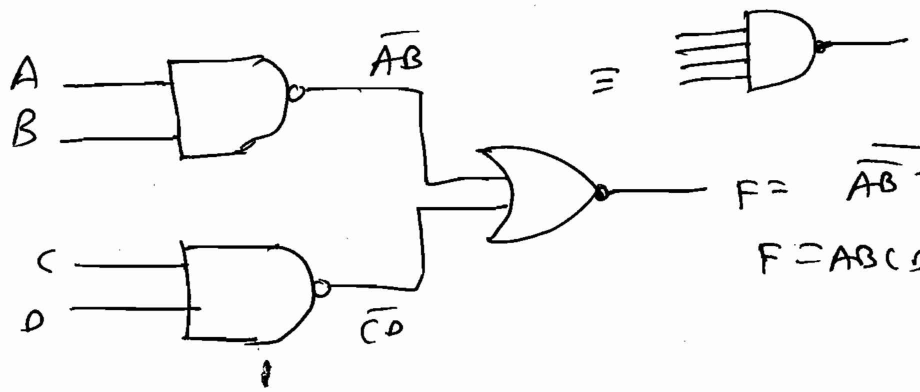
→ ~~Eg.~~ (1) Degenerative

(2) Non-Degenerative.

① Degenerative.

⇒ There are only one logical operation in o/p, then it is called Degenerative type.

e.g. NAND - NOR Logic



$F = \overline{\overline{AB} + \overline{CD}}$

$F = ABCD$

→ The advantages of Degenerative form is the ~~facing~~ fan-in of the gate is increase.

☆ Karnaugh Maps (Veitch Diagram).

* 3- Variable K- Map:

F(A, B, C)

A \ BC	00	01	11	10
0	0	1	3	2
1	4	5	7	6

octet (group of 8 adj minterms).

↓ x

Quad (group of 4 adj minterms).

↓ x

pair (group of 2 adj minterms.

Q: How many possible ways to get ~~[illegible]~~ Quad of minterms?

Ans: 6.

* 4- Variable k-map

AB \ CD	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	12	13	15	14
10	8	9	11	10

Possible Quads => 24

Possible octets => 8.

Possible pair => 32

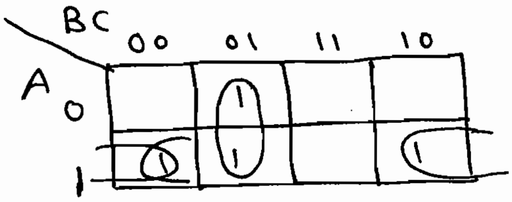
(0, 2, 8, 10) => Quad.

Colums	Rows
✓ 1, 2	1, 2
✓ 2, 3	2, 3
✓ 3, 4	3, 4
✓ 4, 1	4, 1

Ex-1 Simplify the following expression using maping.

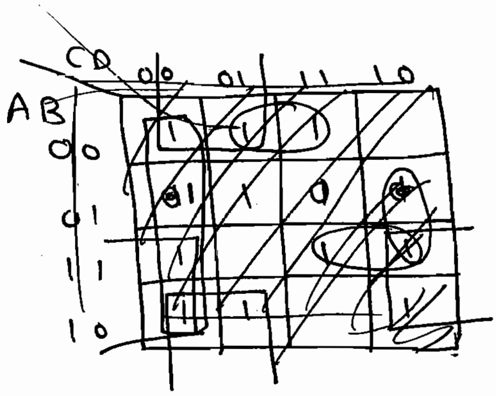
(a) $F(A,B,C) = \Sigma m(1,4,5,6)$

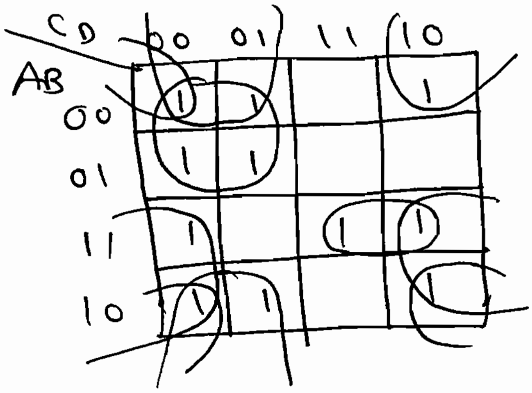
→



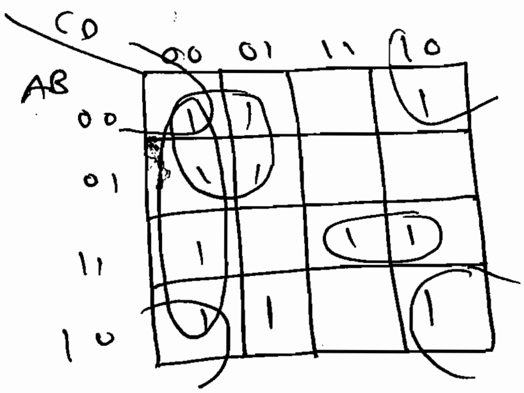
$\therefore F = A\bar{C} + \bar{B}C.$

(b) $F(A,B,C,D) = \Sigma m(0,1,2,4,5,8,9,10,12,14,15).$





(OR)



$\boxed{\text{NOTE}}$ The simplified expression obtained using k-map is $\boxed{\text{minimal}}$ but $\boxed{\text{not unique.}}$

Ex-1 a) $F(A,B,C) = \Sigma m(1,2,4,7)$.

b) $F_1(A,B,C) = \Sigma m(0,3,5,6)$.

Ans: a)

A \ BC	00	01	11	10
0		1		1
1	1		1	

	A	B	C
m_1	0	0	1
m_2	0	1	0
m_4	1	0	0
m_7	1	1	1

$F = A\bar{B}\bar{C} + \bar{A}\bar{B}C + A\bar{B}\bar{C} + ABC.$

$F = \bar{B}(A\bar{C} + \bar{A}C) + A(\bar{B}\bar{C} + BC).$

$\therefore \boxed{F = A \oplus B \oplus C.}$

→ All minterms have odd no. of 1's.

→ All minterms have even no. of 0's.

b) F_1

A \ BC	00	01	11	10
0	1		1	
1		1		1

$F = \Sigma m(1,2,4,7)$

$+\ \bar{F} = \Sigma m(0,3,5,6)$

$F + \bar{F} = 1$

$\therefore F = A \odot B \oplus C.$ $\quad F_1 = \bar{F}$

$F_1 = \overline{A \oplus B \oplus C} = \bar{A} \odot \overline{B \oplus C}$

let, $\bar{z} = A \oplus B$
$= A \odot B$

$\therefore F_1 = \overline{z \oplus C}$

$\boxed{F_1 = A \odot B \oplus C}$. (or)

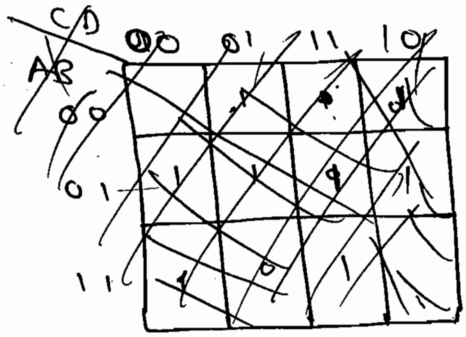
$\boxed{F_1 = A \oplus B \odot C.}$

→ Minterms doesn't have odd no. zeros and even no. of 1's.

Ex-2 Represent $F = A \oplus B \oplus C \oplus D$ in sum of minterms (Canonical SOP form).

Ans:

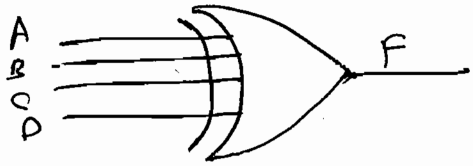
$F = A \oplus B \oplus C \oplus D.$



CD \ AB	00	01	11	10
00		1		1
01	1		1	
11		1		1
10	1		1	

Ans: 8

A	B	C	D	F
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1



$F = \Sigma m(1, 2, 4, 7, 8, 11, 13, 14).$

Hint: $\boxed{\text{All above minterms have odd no. of ones.}}$

Ex-3 Simplify the following map:

$F = \Sigma m(0, 3, 5, 6, 9, 10, 12, 15).$

CD \ AB	00	01	11	10
00	1		1	
01		1		1
11	1		1	
10		1		1

* All minterms contain even no. zeros

So, X-NOR gate.

$\therefore \boxed{F = A \odot B \odot C \odot D.}$

Ex-4 In n-variable k-map a group of 8^{m} is form. How many literals the resulting minterms.

Ans:

→ 3-var k-map ⟶

A \ BC				
	1	1	1	1
	1	1	1	1

F = 1

so, 0 literal

→ 4-var k-map ⟶

AB \ CD	00	01	11	10
00		1	1	
01		1	1	
11		1	1	
10		1	1	

F = D

so, 1 literal.

⋮

→ n-var k-mp → there are n-3 literals. require for 8 minterms.

for pair → n-1 literals.

→ for Quad → n-2 literals.

for ~~8 min~~ octet → n-3 literals.

Ex-5 Determine the simplified pos expression of F where, $F = \sum m(0, 2, 3, 4, 8, 10, 11, 14)$.

Ans: for pos.

$\therefore F = \prod M(1, 5, 6, 7, 9, 12, 13, 15)$.

for max term

0 → var

$\overline{\text{var}}$

AB \ CD	00	01	11	10
00		0		
01		0	0	0
11	0	0	0	
10		0		

$F = (\overline{B} + \overline{D}) \cdot (C + \overline{D}) \cdot (\overline{A} + \overline{B} + \overline{C})(A + \overline{B} + \overline{C})$.

* Implicant:

→ (i) Prime implicant &

(ii) Essential implicant

→ **Implicant:**

→ it is the set of all adjecent minterms.

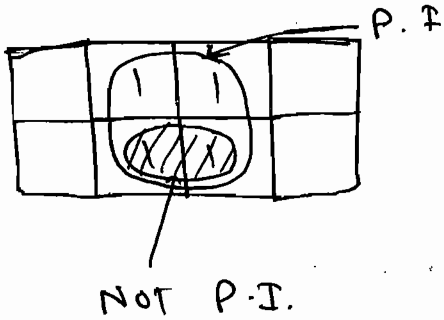
E.g. octents, pairs.

* Prime implicant:

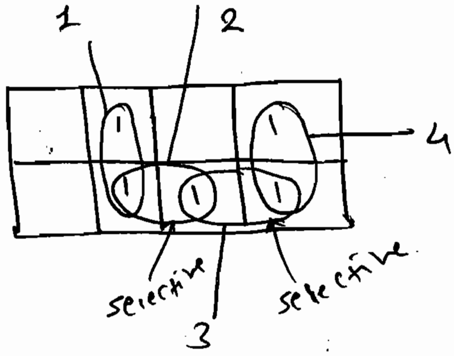
→ It is an implicant which is not a subset of another implicant.

e.g.

(a)

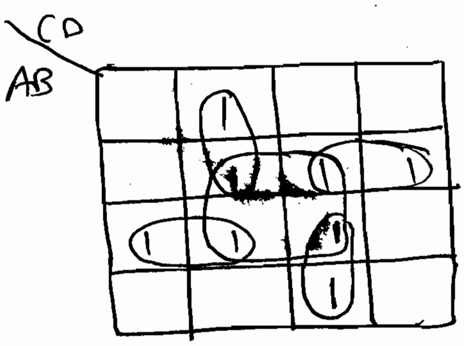


(b)



All 4 are prime impilcants.

(c)

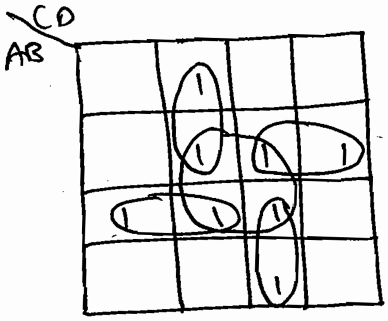


All are prime implicants.

* Essential Prime Implicants:

→ It is a prime implicant which contains atleast 1 minterms which is not covered by another prime implicant.

e.g.



← is non-essential prime implicant and remaining are essential prime implicant.

* Non-Essential Prime Implicants:

1) Redundant P.I (RPI).

→ It is a non-essential prime implicant whose minterms are covered by all essential P.I.

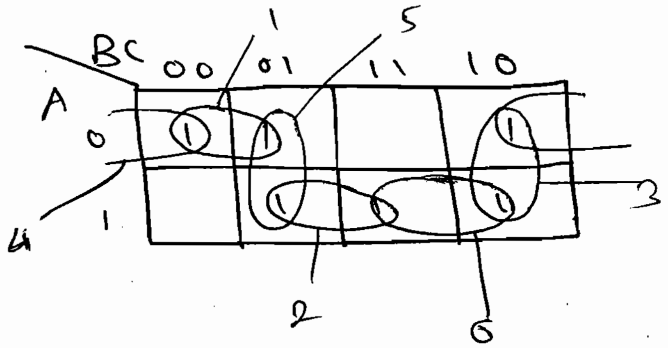
2) Selective Prime implicant:

→ It is a non-essential prime implicant whose minterms are covered by atleast one non-essential P.I.

* $\boxed{\text{Minimal Expression} = \text{EPI's} + \text{(optional) SPI}}$

Ex-1 Determine the essential P.I. and minimal expression for the following f^ns.

Ans: (1) $F(A,B,C) = \Sigma m(0,1,2,5,6,7)$.



EPI's → Null

RPI's → Null

SPI's → 1, 2, 3, 4, 5, 6

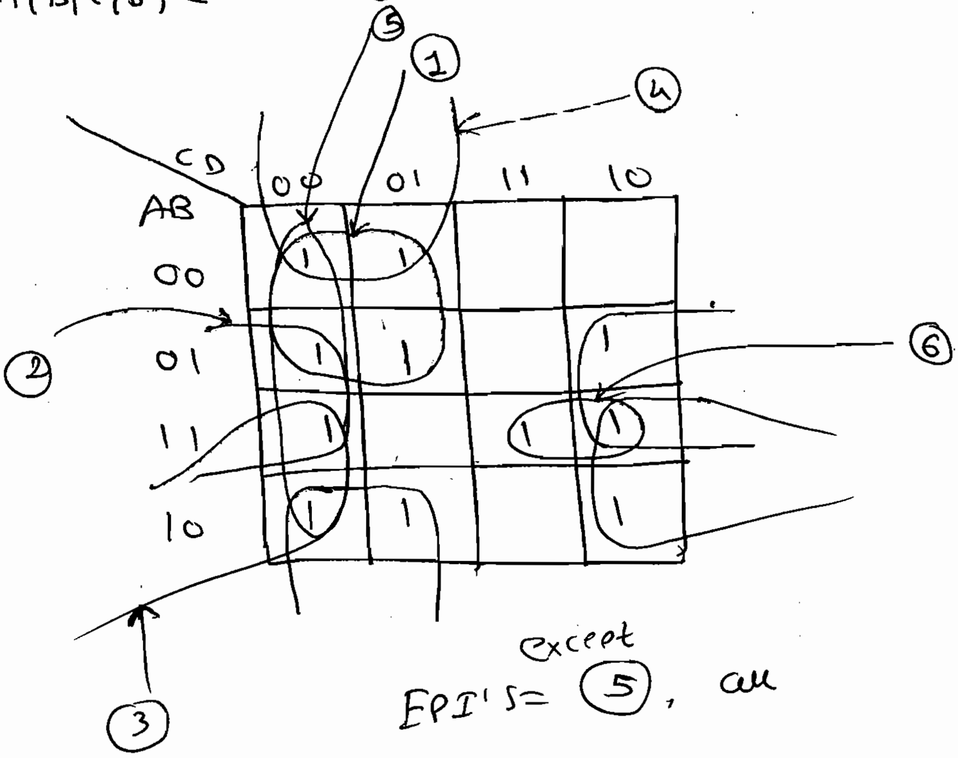
(~~term~~)

minimal expression = 1 + 2 + 3

(or)

= 4 + 5 + 6.

(b) $F(A,B,C,D) = \Sigma m(0,1,4,5,6,8,9,10,12,14,15)$



EPI's = except 5, all

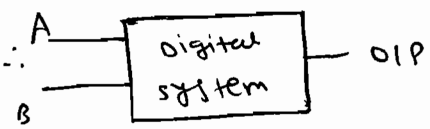
$\therefore F = 1 + 2 + 3 + 4 + 6$.

$F = \bar{A}\bar{C} + B\bar{D} + A\bar{D} + \bar{C}\bar{B} + ABC$

☆ Don't Care Condition:

→ In a digital system for a non-occuring i/p, the o/p can be consider as either 0 (OR) 1 during its simplification and it is called the don't care condition.

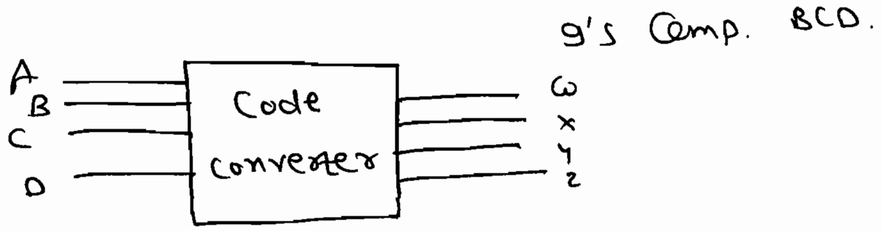
$F(A,B) = \Sigma m(0,2) + d(3).$



AB	F
00	1
01	0
10	1
11	$\boxed{1/0}$ ← don't care condition.

Ex-1 Design BCD to 9's Comp. of BCD Code converter.

Ans:



$\boxed{\begin{array}{c} 0000 \\ \vdots \\ 1001 \end{array}}$

Don't care $\left\{ \begin{array}{l} 1011 \rightarrow x \\ \vdots \\ 1111 \rightarrow x \end{array} \right.$

	BCD				9's Comp. of BCD			
	A	B	C	D	W	X	Y	Z
0	0	0	0	0	1	0	0	1
1	0	0	0	1	1	0	0	0
2	0	0	1	0	0	1	1	1
3	0	0	1	1	0	1	1	0
4	0	1	0	0	0	1	0	1
5	0	1	0	1	0	1	0	0
6	0	1	1	0	0	0	1	1
7	0	1	1	1	0	0	1	0
8	1	0	0	0	0	0	0	1
9	1	0	0	1	0	0	0	0

(i) for W:

AB \ CD	00	01	11	10
00	1	1		
01				
11				
10				

$W = \bar{A}\bar{B}\bar{C}$

(ii) for X:

$X = \Sigma m(2,3,4,5) + d(10,11,\dots 15)$

AB \ CD	00	01	11	10
00			1	1
01	1	1		
11	X	X	X	X
10			X	X

$X = B\bar{C}$

(iii) $Y = C$

direct from T.T

(iv) $Z = \bar{D}$

direct from T.T

*** NOTE:**

→ The purpose of minimization is to reduce the number of logic gates and no. of inputs.

Ex-1 (i) $F_1(A,B,C) = \Sigma m(0,2,3,5) + d(1,6,7)$.

(ii) $F_2(A,B,C) = \Sigma m(0,1,2,3,6) + d(4,5,7)$.

Find $F_3 = F_1 + F_2$

$F_4 = F_1 \cdot F_2$.

Ans: (i) $\Sigma m(0,2,3,5) + d(1,6,7)$.

A \ BC	00	01	11	10
0	1	X	1	1
1		1	X	X

$F_1 = \bar{A} + C + B$.

(ii) $\Sigma m(0,1,2,3,6) + d(4,5,7)$.

A \ BC	00	01	11	10
0	1	1	1	1
1	X	X	X	1

$F_2 = 1$.

$\therefore F_3 = F_1 + F_2 = 1 + (\bar{A}+C) = 1$

$F_4 = F_1 \cdot F_2 = F_1 = \bar{A} + C$

(OR)

$1 + d = 1$	$0 \cdot d = 0$
$0 + d = d$	$1 \cdot d = d$
$d + d = d$	$d \cdot d = d$.

$F_3 = F_1 + F_2 = \sum m(0, 1, 2, 3, 5, 6) + d(4, 7)$

$F_4 = F_1 \cdot F_2 = \sum m(0, 2, 3) + d(1, 6, 7, 5)$.

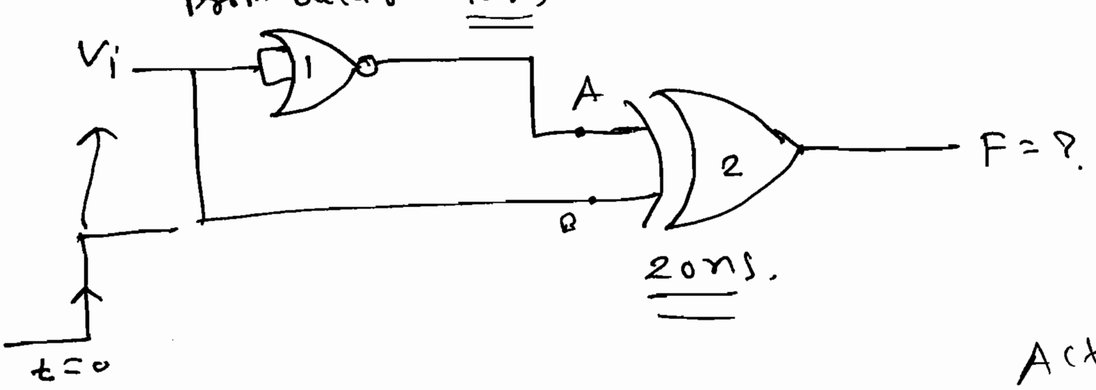
Ex-2 $F_1(A, B, C) = \sum m(0, 3, 5, 6) + d(2, 4, 7)$.

$F_2(A, B, C) = \sum m(1, 2, 3, 6) + d(0, 5, 7)$.

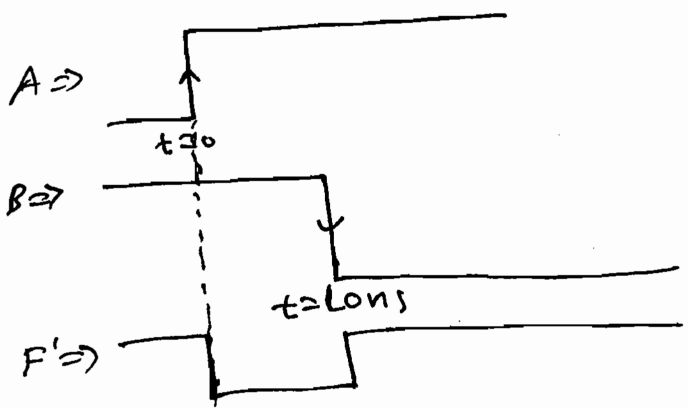
Ans: $F_3 = F_1 + F_2 = \sum m(0, 1, 2, 3, 5, 6) + d(4, 7)$.

$F_4 = F_1 \cdot F_2 = \sum m(3, 6) + d(0, 2, 5, 7)$.

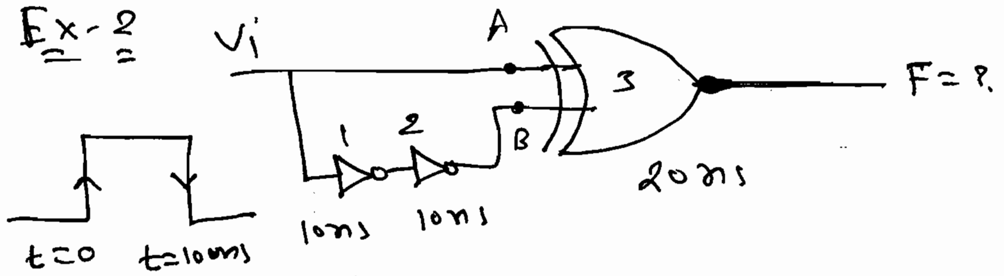
Ex-3 Determine the waveform at the o/p of the following logic ckt.



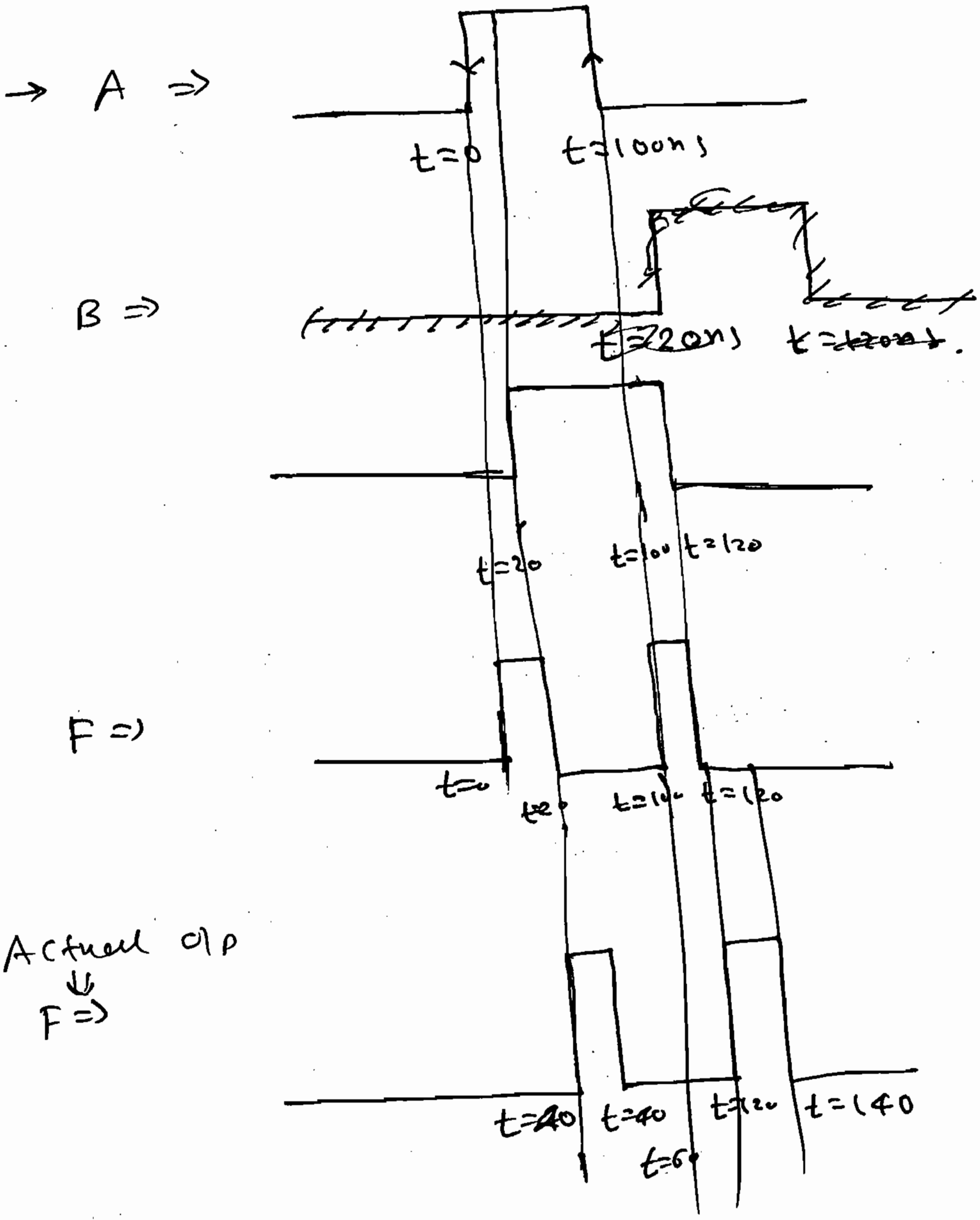
Actual o/p.



Ex-2



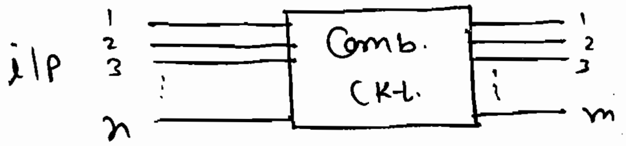
Vi
A
B
1
2
3
10ns
10ns
20ns
F = ?
t=0
t=100ns



→ A ⇒
t=0
t=100ns
B ⇒
t=20ns
t=120
t=20
t=100
t=120
F ⇒
t=0
t=20
t=100
t=120
Actual o/p
F ⇒
t=40
t=40
t=60
t=120
t=140

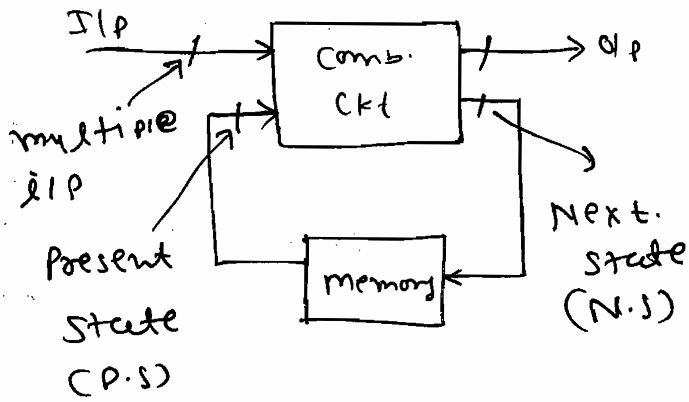
☆ Digital circuits

- Combinational ckt
- Sequential ckt.



→ Outputs = F (Present I/P)

E.g. = Code Converter,
Adders & Subtractor,
Decoder, MUX,
Mag. Comparators, PLA,
$\boxed{\text{Rom}}$



→ <u>Mealy seq. ckt.</u>

O/P = F (present state, present inputs)

Next state = F (present state, present i/p.)

→ <u>Moore</u> <u>seq</u> <u>ckt.</u>

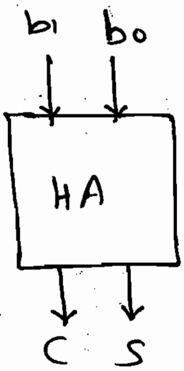
O/P = F (present state only)

N.S. = F (present Inputs, present state).

E.g. Shift register, Counters, calculators, MP, PC.

* Arithmetic Combinational Ckt.

(1) Half Adder:



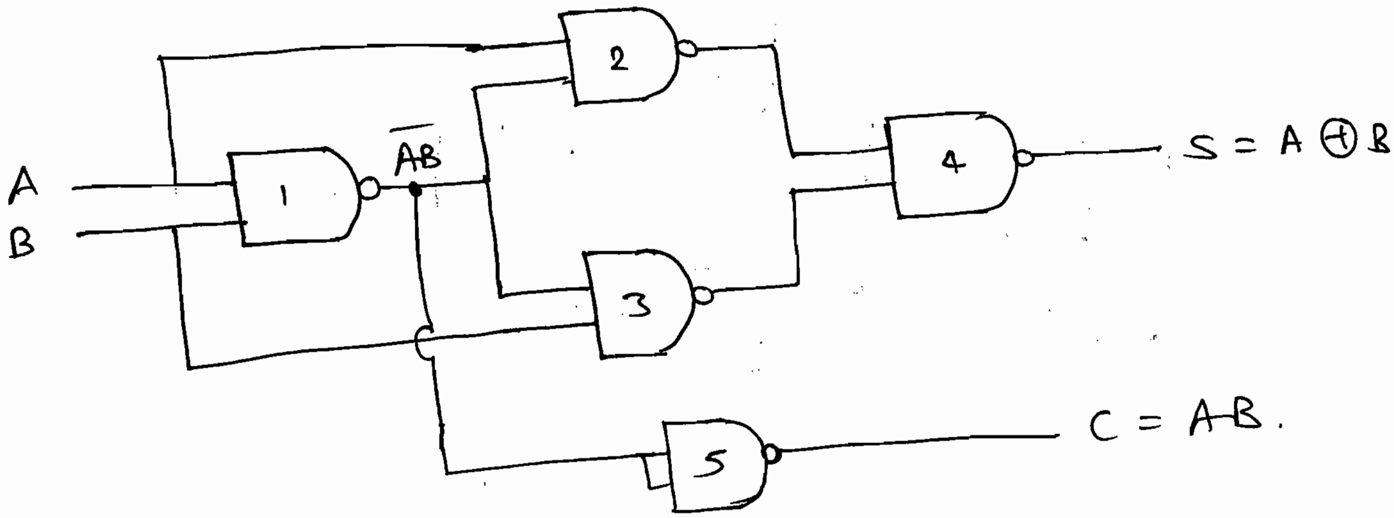
A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$S = \bar{A}B + A\bar{B}$

$S = A \oplus B.$

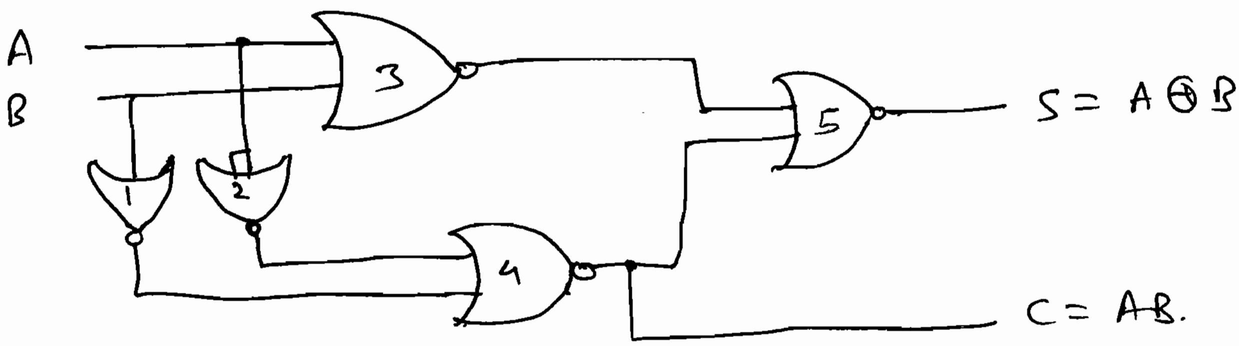
$C = A \cdot B.$

⇒ Half Adder using NAND Gates.



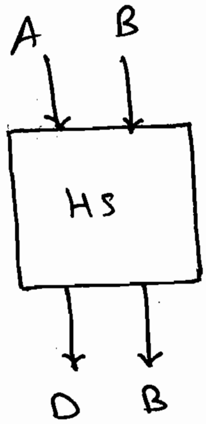
So, 5 NAND gate required

⇒ Half Adder using NOR gate



So, 5 NOR gate required

(2) Half Subtractor:



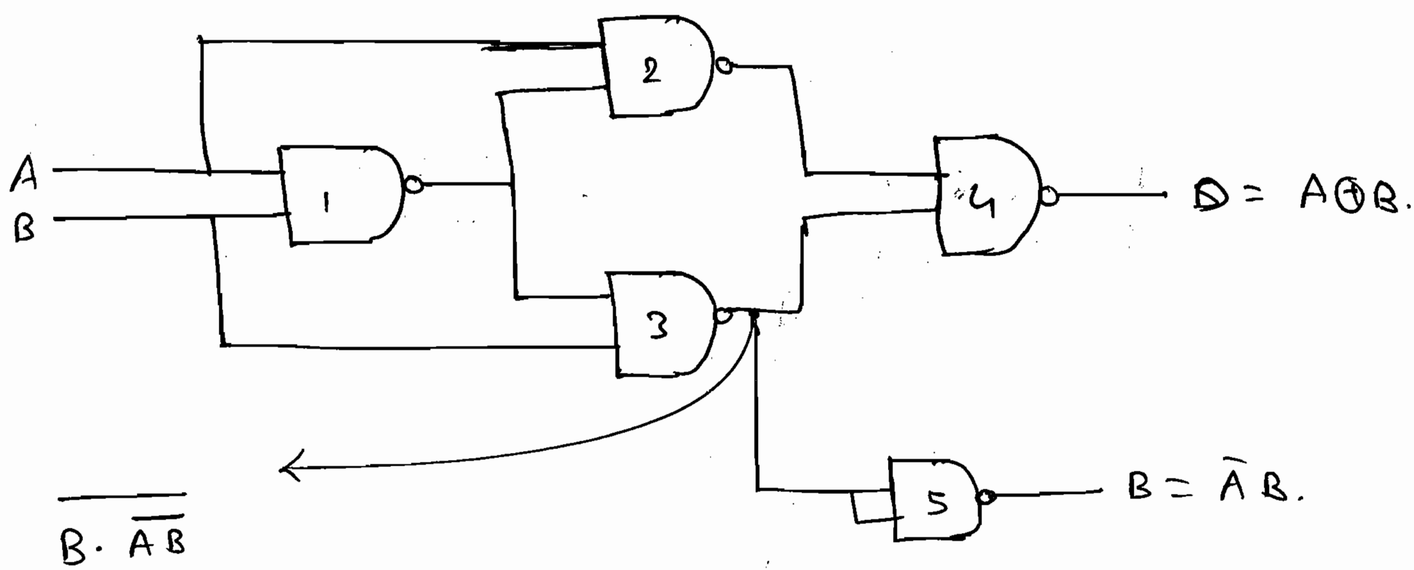
A	B	D	B
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

$D = A \oplus B$

$B = \overline{A} \cdot B.$

$$\begin{array}{r} B \leftarrow 1\ \cancel{0}^{2} \\ -\ 1 \\ \hline 1 \rightarrow D \end{array}$$

⇒ Half ~~Adder~~ Subtractor using NAND gate only:



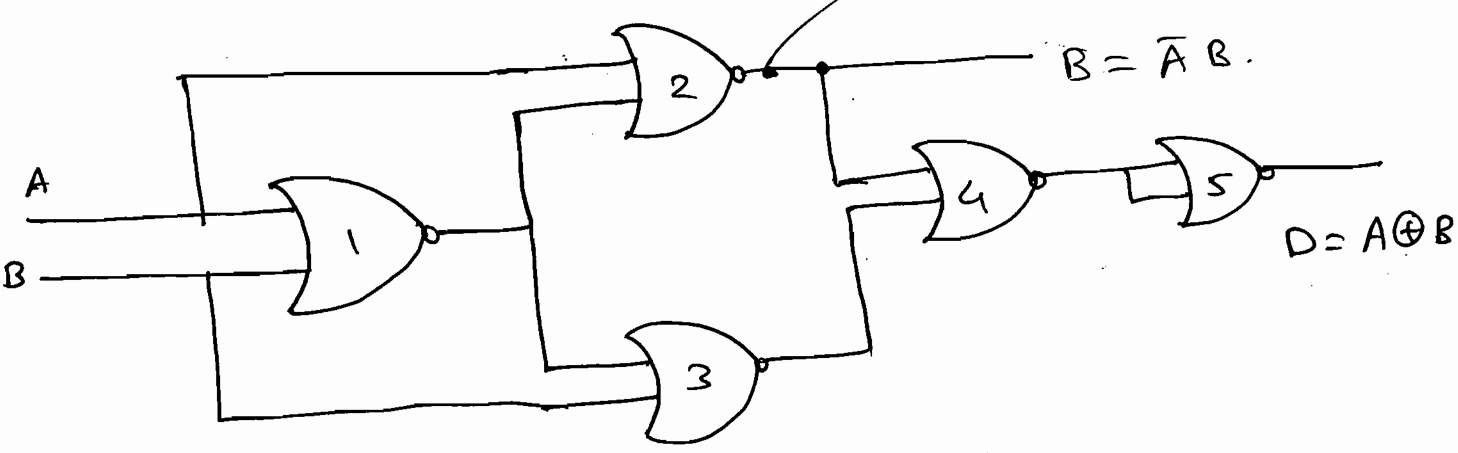
$\overline{B \cdot \overline{AB}}$

$= \overline{B} + AB$

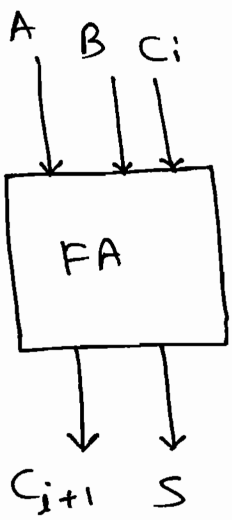
$= A + \overline{B} \quad = \overline{\overline{A} \cdot B}$

⇒ Half Subtractor using NOR gate only.

$$\overline{A+\overline{A+B}} = \bar{A}\cdot(A+B)$$
$$= \bar{A}B$$



(3) Full Adder: (1 bit adder).



	A	B	C_i	S	C_{i+1}
	0	0	0	0	0
	0	0	1	1	0
Carry Propogation	0	1	0	1	0
	0	1	1	0	1
	1	0	0	1	0
	1	0	1	0	1
Carry Generation	1	1	0	0	1
	1	1	1	1	1

$$S(A, B, C_i) = \Sigma m(1, 2, 4, 7)$$

A \ BC_i	00	01	11	10
0		1		1
1	1		1	

$$S = A \oplus B \oplus C_i \text{ (or) } A \odot B \odot C_i$$

$$C_{i+1} = \Sigma m(3, 5, 6, 7)$$

$$= \bar{A}BC_i + A\bar{B}C_i + AB\bar{C_i} + ABC_i$$

$$= C_i(\bar{A}B + A\bar{B}) + AB.$$

$$\therefore \boxed{C_{i+1} = C_i(A \oplus B) + AB}$$

A \ BC_i	00	01	11	10
0			1	
1		1	1	1

$$\boxed{C_{i+1} = AB + BC_i + AC_i}$$

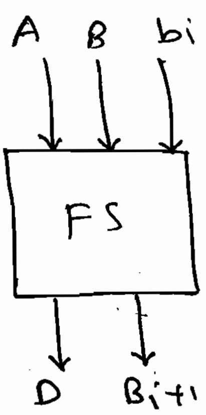
NOTE: Full adder

(i) Require 9 NAND gate

(ii) Require 12 NOR gate.

(4) Full Subtractor:

→



A	B	bi	D	bi+1
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

$\rightarrow$ $D(A, B, b_i) = \sum m(1, 2, 4, 7)$.

A \ Bb_i	00	01	11	10
0		1		1
1	1		1	

$$D = A \oplus B \oplus b_i \quad \text{(or)} \quad A \odot B \odot b_i.$$

$\rightarrow$ $b_i(A, B, b_i) = \sum m(1, 2, 3, 7)$.

$$b_{i+1} = m_1 + m_2 + m_3 + m_7$$

$$= \bar{A}\bar{B}b_i + \bar{A}B\bar{b_i} + \bar{A}Bb_i + ABb_i$$

$$= b_i(\bar{A}\bar{B} + AB) + \bar{A}B.$$

$$b_{i+1} = b_i(A \odot B) + \bar{A}B.$$

A \ Bb_i	00	01	11	10
0		1	1	1
1		0	1	0

$$\therefore \boxed{b_{i+1} = \bar{A}B + Bb_i + \bar{A}b_i}$$

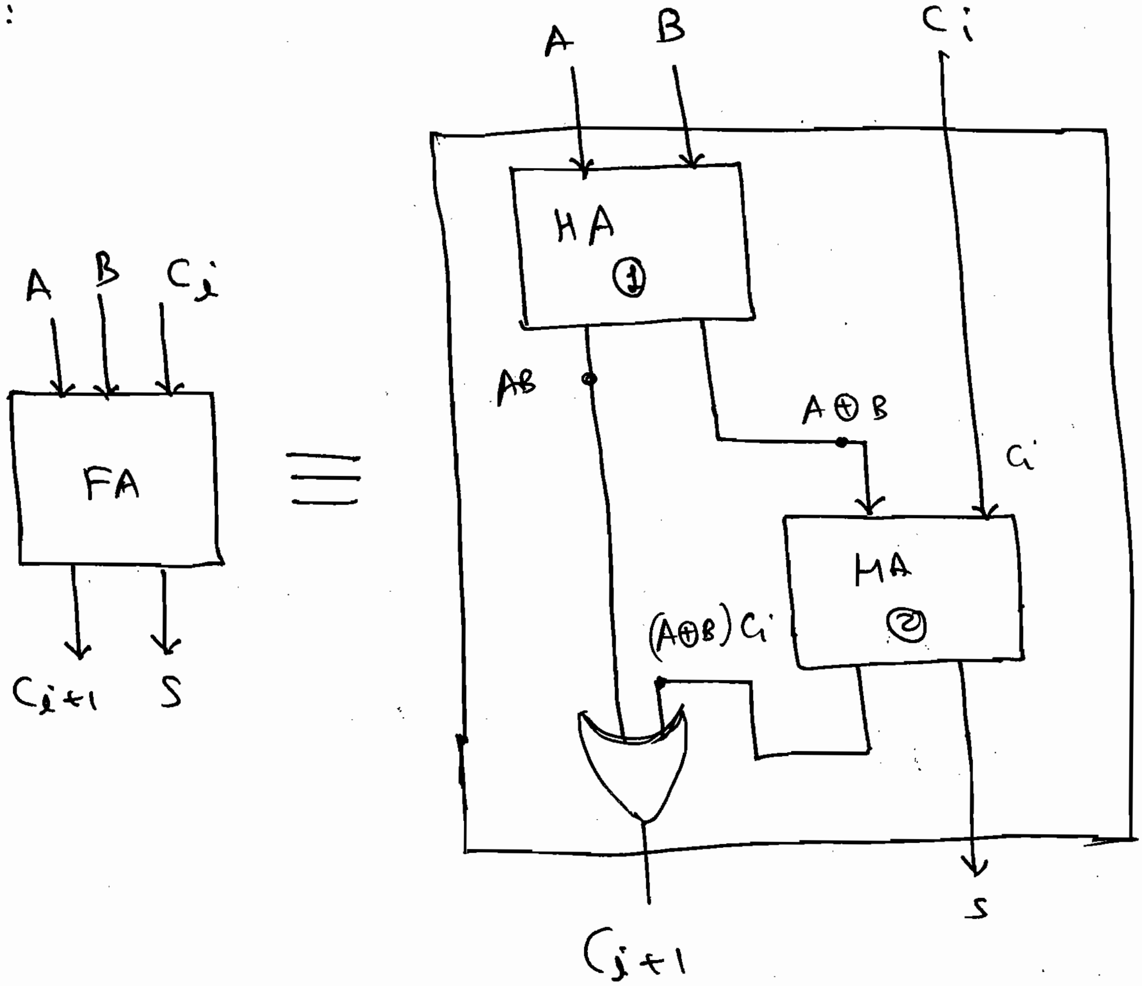
NOTE: Full subtractor

(i) Required 9 NAND gate

(ii) Required 12 NOR gate.

Ex-1 Implement Full Adders using HAs and Logic gate:

Ans:



$= AB + C_i(A \oplus B)$

1 Full adder = 2 half adder + 1 OR gate

Ex-2 How many HA required to implement the following f^n.s.

$F_1 = \bar{A}C + AB\bar{C} + \bar{B}C.$

$F_2 = A \oplus B \oplus C.$

$F_3 = \bar{A}BC + A\bar{B}C.$

Ans: (i) $F_1 = \bar{A}C + AB\bar{C} + \bar{B}C.$

$= \overline{AB} \cdot C + AB\bar{C}$

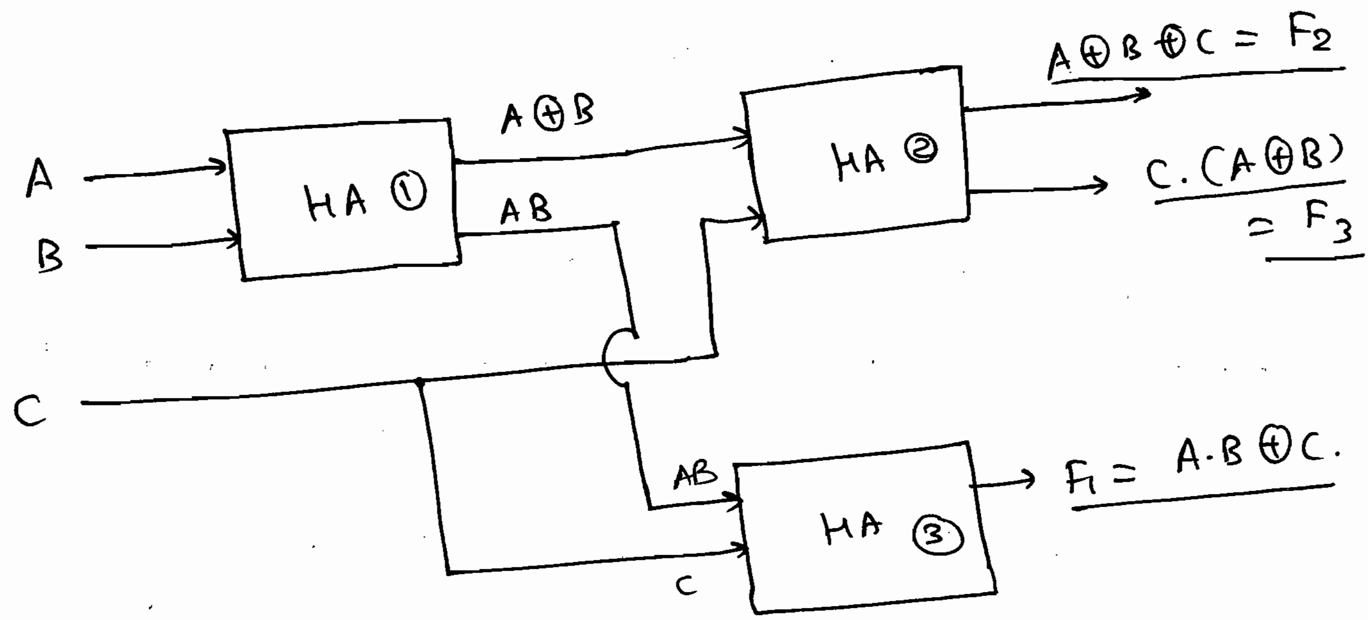
$\cancel{(A \oplus B) \cdot C}$ $\qquad X = \overline{AB}.$

$= \bar{X} \cdot C + X\bar{C}$

$\boxed{F_1 = A \cdot B \oplus C} \longrightarrow$

(ii) $\boxed{F_2 = A \oplus B \oplus C}.$

(iii) $\boxed{F_3 = (\bar{A}B + A\bar{B})\,C = C \cdot (A \oplus B)}$



* Binary Adders:

(1) Parallel binary Adder:

(2) Carry Look Ahead Adder:

(3) Serial Adder.

(1) Parallel binary Adder: (4 bit):

→ ~~RPA~~

[$\boxed{IC = 74LS83}$]

→

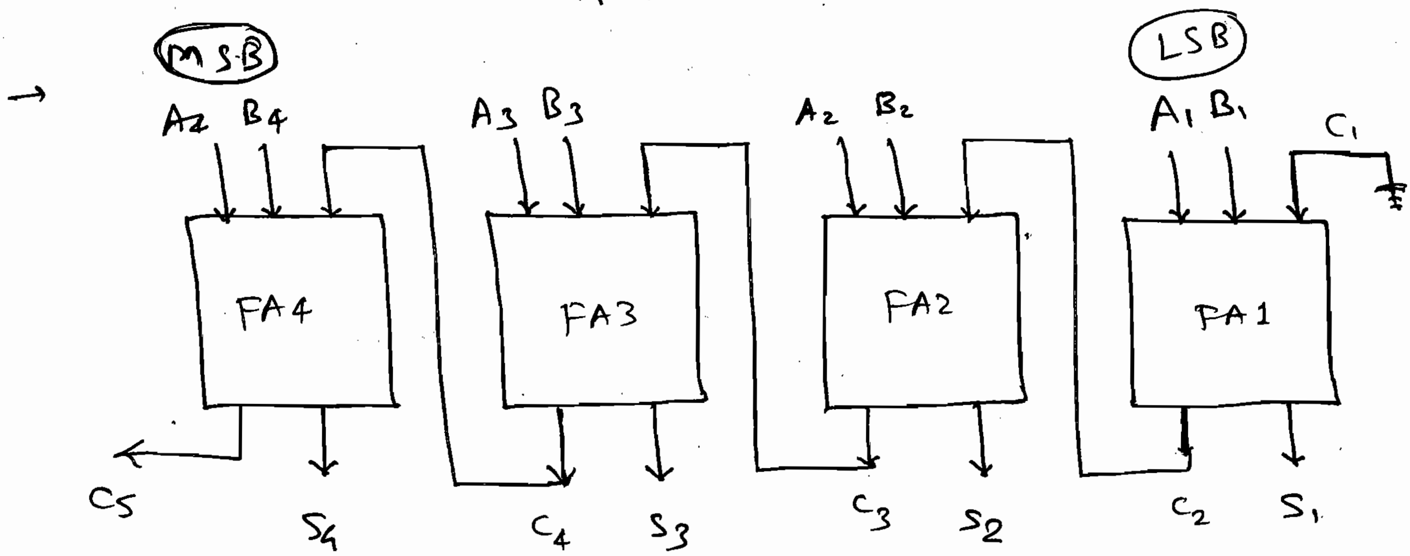
$$\begin{array}{rcccc} A = & \underbrace{A_4 \quad A_3 \quad A_2}_{FA} & & & A_1 \\ + & B_4 \quad B_3 \quad B_2 & & & B_1 \\ \hline C_5 & S_4 \quad S_3 \quad S_2 & & & S_1 \end{array}$$

HA (OR) FA

~~RPA~~ → 3 FA + 1 HA (OR)

4 FAs

→



Q: how many HA required to implement 4-bit full adder

Ans: 7 HA &

3 OR gate

Q.

→ In a 4 bit Parallel binary adder a FA takes 32 ns to produce the sum and 14 ns to produce the carry. Determine

(i) Time required for addition.

(ii) the adition rate of the adder.

Ans: a)

Time required for Addition in N-Bit parallel adder $= T = (N-1)t_c + \max(t_s, t_c)$.

$\Rightarrow T = (4-1)14 + \max(32, 14)$.

$= 42 + 32$

$\boxed{T = 74\,ns}$

b) $\text{Addition Rate} = \frac{1}{T} = \frac{1}{74 \times 10^{-9}}$ Hz.

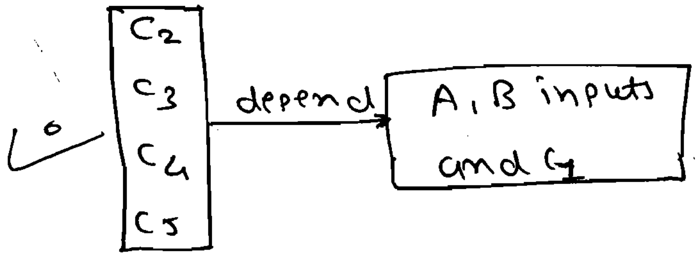
→ This can be used upto 4 bit.

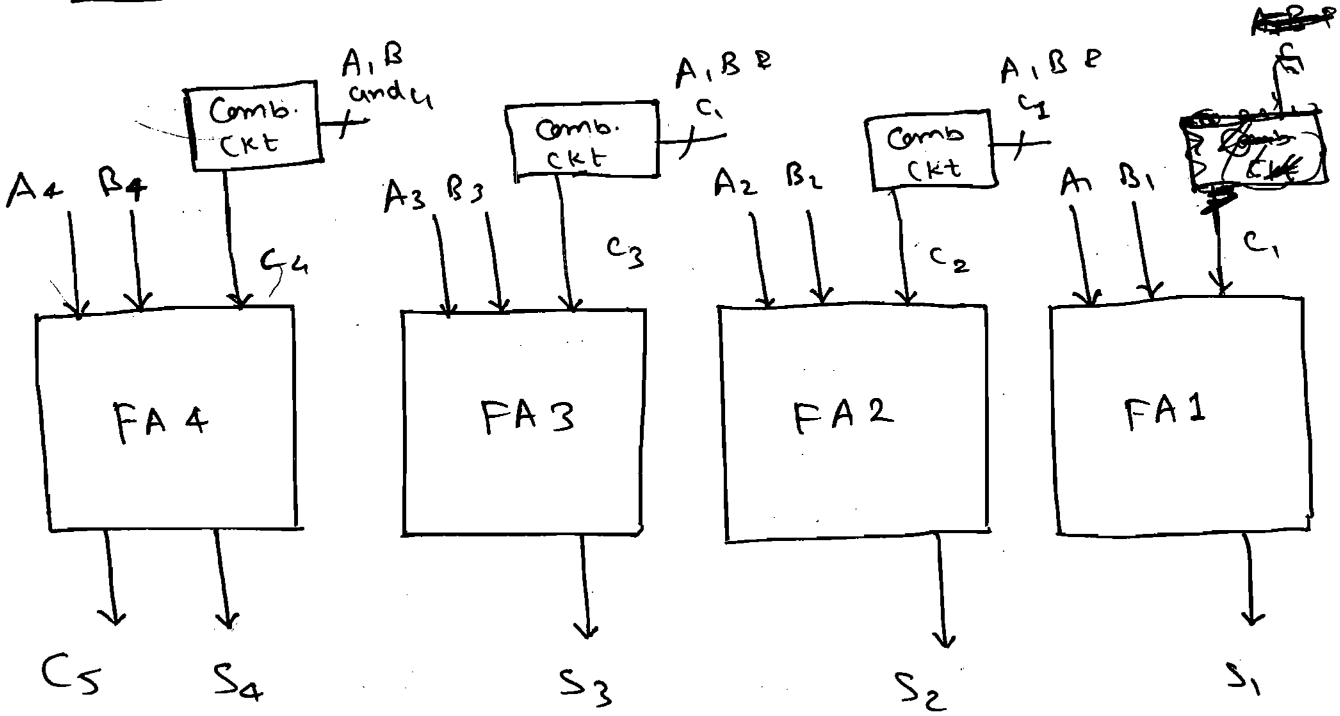
→ Disadvantages:

→ as the size of the Adder increases the Speed operation decreases as the carry has to propogate through all the FAs to overcome this we use Carry Look ahead Adder.

(2) Carry Look Ahead Adders:

→ ⊛ Principle:





→ $C_{i+1} = C_i\,(A \oplus B) + AB$

$P_i = A \oplus B =$ Propogation of carry

$G_i = AB =$ Carry Generation

$\therefore\ C_{i+1} = C_i P_i + G_i$

→ $\boxed{C_2 = C_1 P_1 + G_1}$ — ①

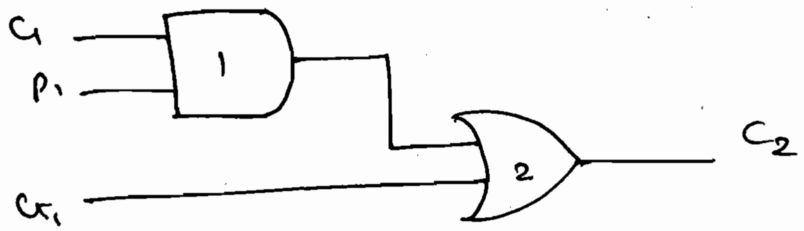
→ $C_3 = C_2 P_2 + G_2$

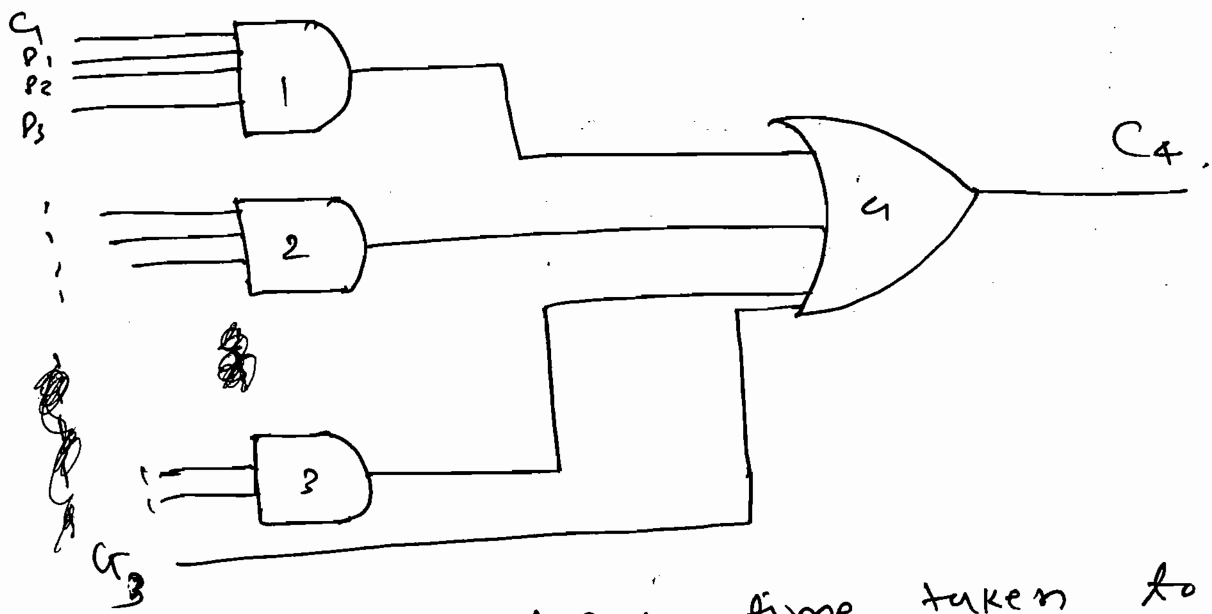
$\boxed{C_3 = C_1 P_1 P_2 + G_1 P_2 + G_2}$ — ②

→ $C_4 = C_3 P_3 + G_3$

$$\rightarrow \boxed{C_4 = C_1P_1P_2P_3 + G_1P_2P_3 + G_2P_3 + G_3} \quad — (3)$$

*





→ In Carry Look-Ahead time taken to generate the carries C_2, C_3, C_4 is same as they are implimented by 2 level Logic.

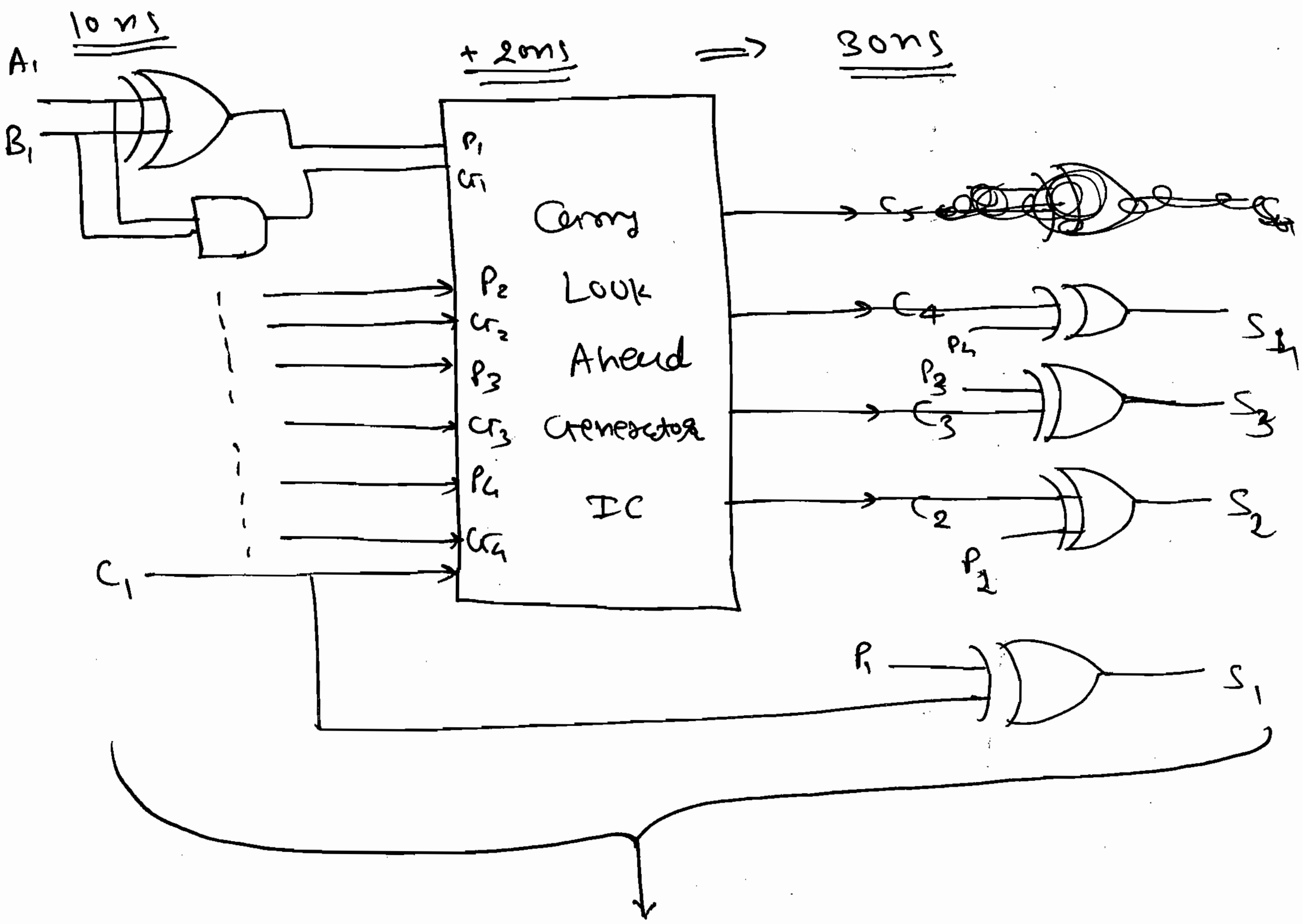
* Advantage:

→ Its Speed of operation is very high and doesn't depend on the size of the adder.

* Disadvantage:

→ It has more hardware Complexity. To overcome this we use carry look ahead generator IC.

⇒ Carry Look Ahead generator IC



Carry Look Ahead Adder

$$S_i = A_i \oplus B_i \oplus C_i$$

$$S_i = P_i \oplus C_i$$

$$S_1 = P_1 \oplus C_1$$

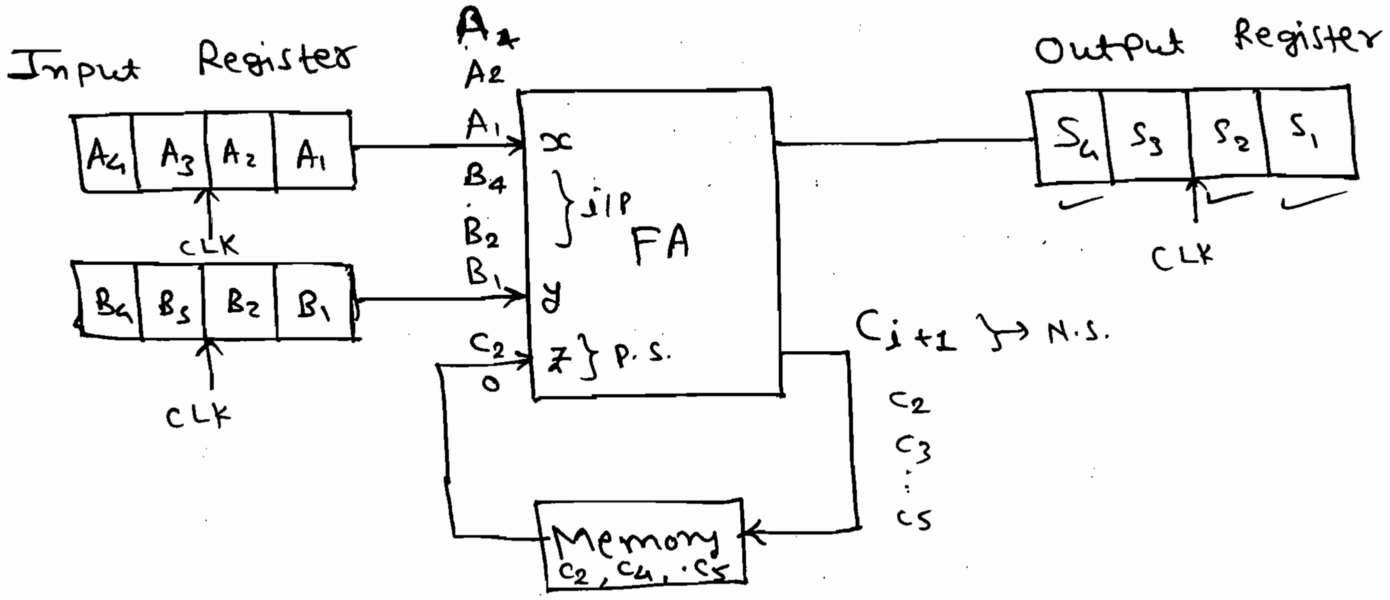
$$S_2 = P_2 \oplus C_2$$

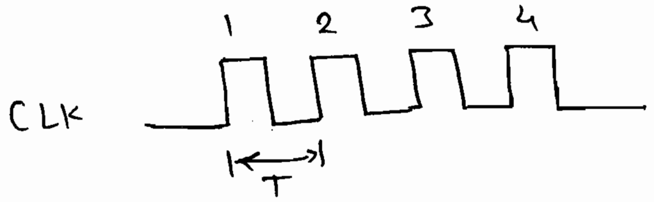
$$S_3 = P_3 \oplus C_3$$

$$S_4 = P_4 \oplus C_4.$$

(3) Serial Adder: (sequential ckt)

→

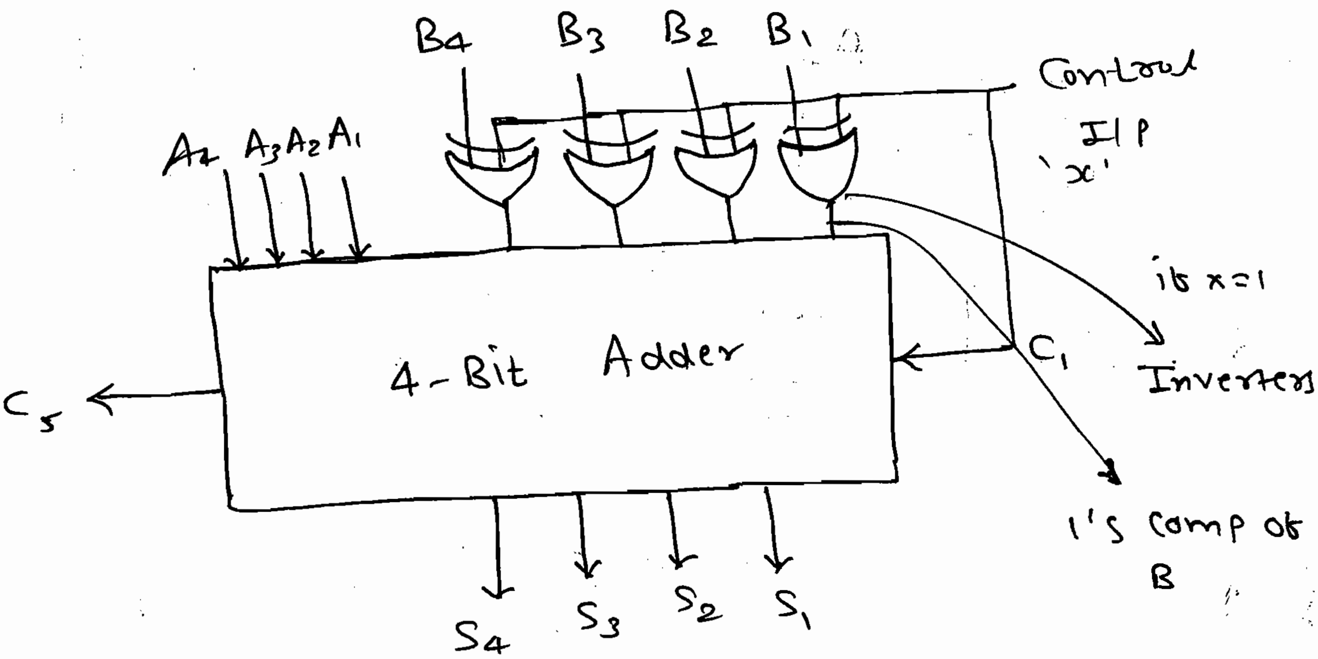




T = CLK period

$$\boxed{\text{Time for 4-bit serial addition} = 4T}$$

* 4-bit Parallel Adder / Subtractor.



(1) If $\boxed{x=1}$ $\Rightarrow$ Ex-OR gate $\rightarrow$ Inverters

$\rightarrow$ A + 1's Comp of B + $(C_1 = 1)$

$\rightarrow$ A + (2's comp of B)

$\Rightarrow$ $\boxed{A - B}$ $\rightarrow$ $\boxed{\text{Binary Subtraction}}$

(2) If $\boxed{x=0}$ $\Rightarrow$ Ex-OR gate $\Rightarrow$ Buffer

$\rightarrow$ A + B + $(C_1 = 0)$

$=$ $\boxed{A + B}$ $\rightarrow$ $\boxed{\text{Binary Adding.}}$

Ex-1 Determine the f^n of the above circuit if the inputs are as given below:

(a) $x = 1$; $A =$ Ex-3 code ; $B = 0011 \Rightarrow$ Function = ?

$\Rightarrow A - B = \text{Ex-3} - 0011$

$= \text{Ex-3} + (1100) + 1$

$=$ BCD code.

Ex-3 to BCD code converter.

$\therefore$ let, Ex-3 of 4 $\rightarrow$ 0111 $\Rightarrow$ 0100 $\rightarrow$ BCD
↑ Ex-3

$$\begin{array}{r} 0111 \\ +\ 1100 \\ +\ \ \ \ \ 1 \\ \hline \cancel{1}\ 0100 \end{array} \quad (\because C = 1)$$

$\Rightarrow 4_{10} = (0100)$ BCD.

(b) $x = 1$, $A = 1001$; $B =$ BCD $\Rightarrow$ Function = ?

Ans: $A = 1001$. | let, $B =$ BCD of 3
$B = 0011$.

$\therefore A - B =$

$$\begin{array}{r} 1001 \\ +\ 1100 \\ +\ \ \ \ \ 1 \\ \hline 1\ 0110 \end{array}$$

$= 6_{10}$ which is 9's comp. of 3_{10}

$\therefore$ So, BCD to 9's comp. of BCD.

$A +$ 1's comp of $B + C_1 = 1$.

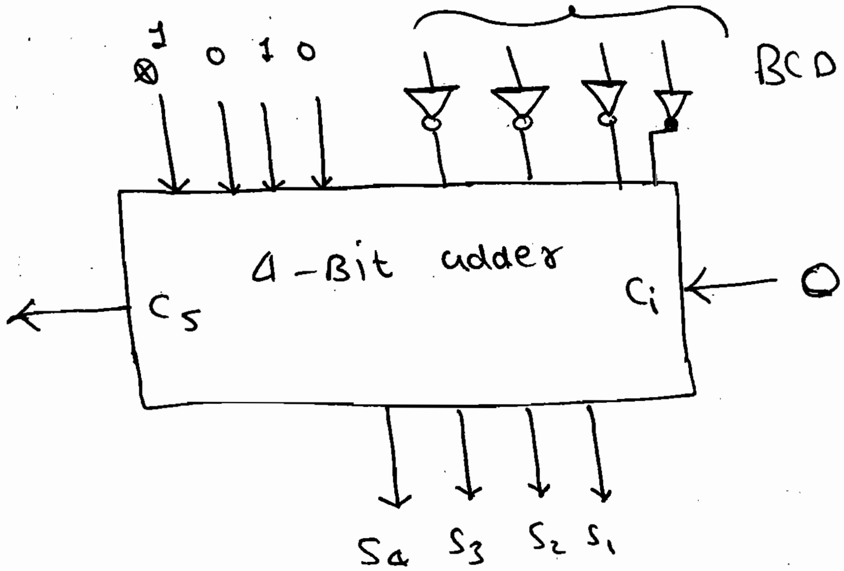
$= A +$ (2's comp of B).

$= A -$ BCD.

$= 9_{10} -$ BCD

$=$ 9's complement of BCD.

(c)



⇒ 1010 + 1's Comp. of BCD $\boxed{C_1 = 0}$

= ~~1010~~ 1001 + 1 + 1's Comp of BCD

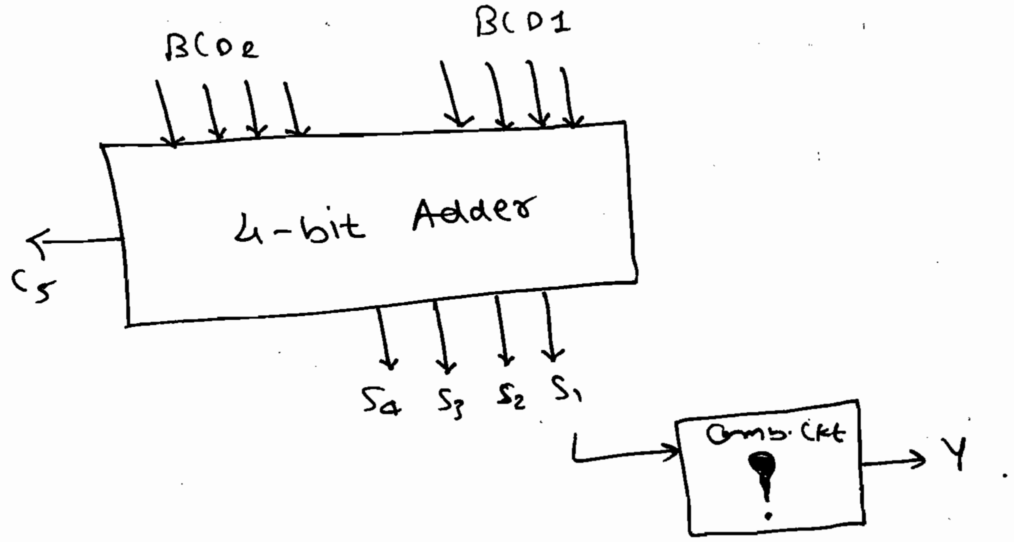
⇒ 1001 + 2's Comp. of BCD.

= 9_{10} − BCD.

→ 9's complement of ckt.

NOTE: The above 9's Complement ckt can be converted <u>to a 10's Complement</u> either by choosing <u>$C_1 = 1$</u> (OR) by choosing A value as <u>A = 1011</u>

* BCD Adder



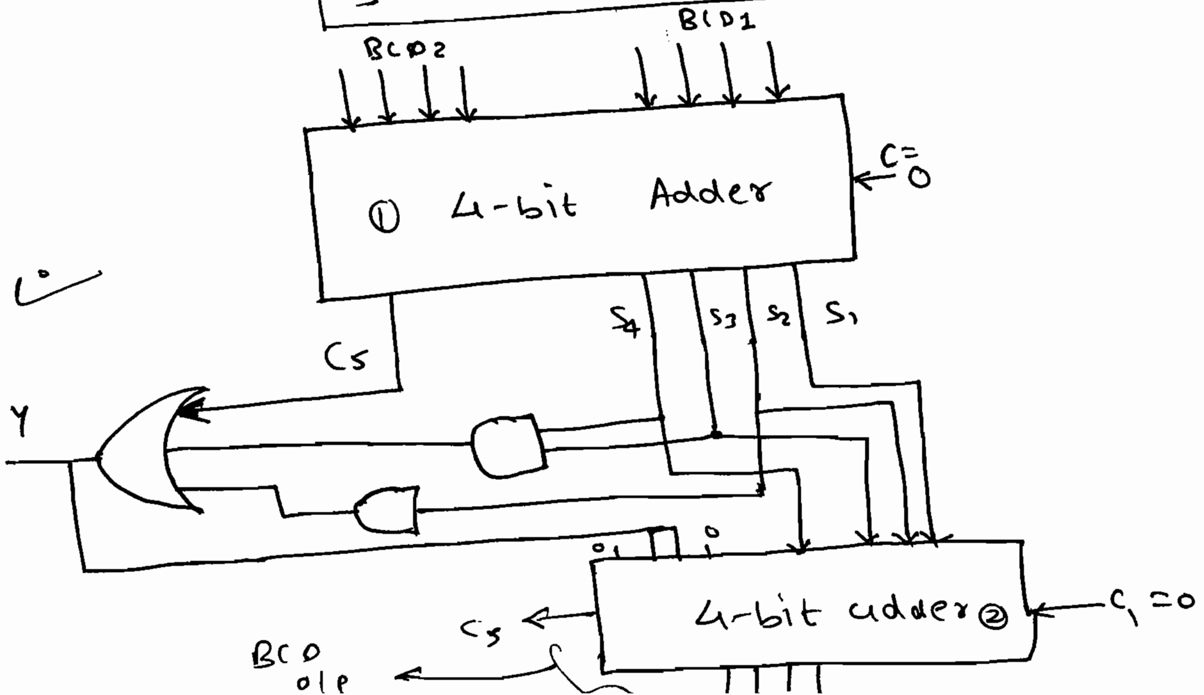
* Output is invalid BCD if

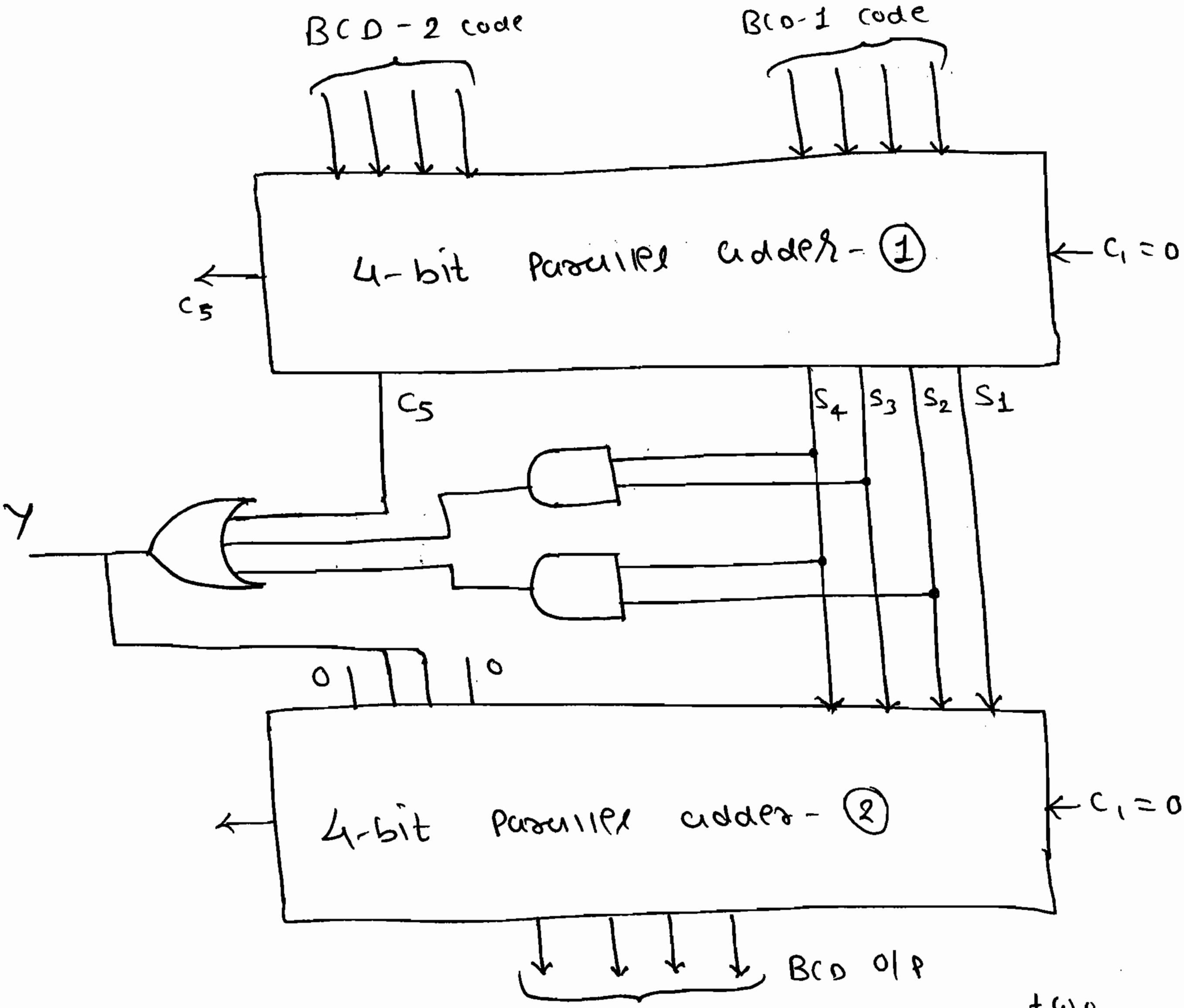
$$\boxed{S_4 S_3 S_2 S_1 > 9 \text{ (or) } C_5 = 1.}$$

$S_4S_3 \backslash S_2S_1$	00	01	11	10
00				
01				
11	1	1	1	1
10			1	1

→ i.e. output is invalid if

$$\boxed{Y = S_4 S_3 + S_4 S_2 + C_5.}$$





→ Hence, one BCD adder requires two 4-bit parallel adders

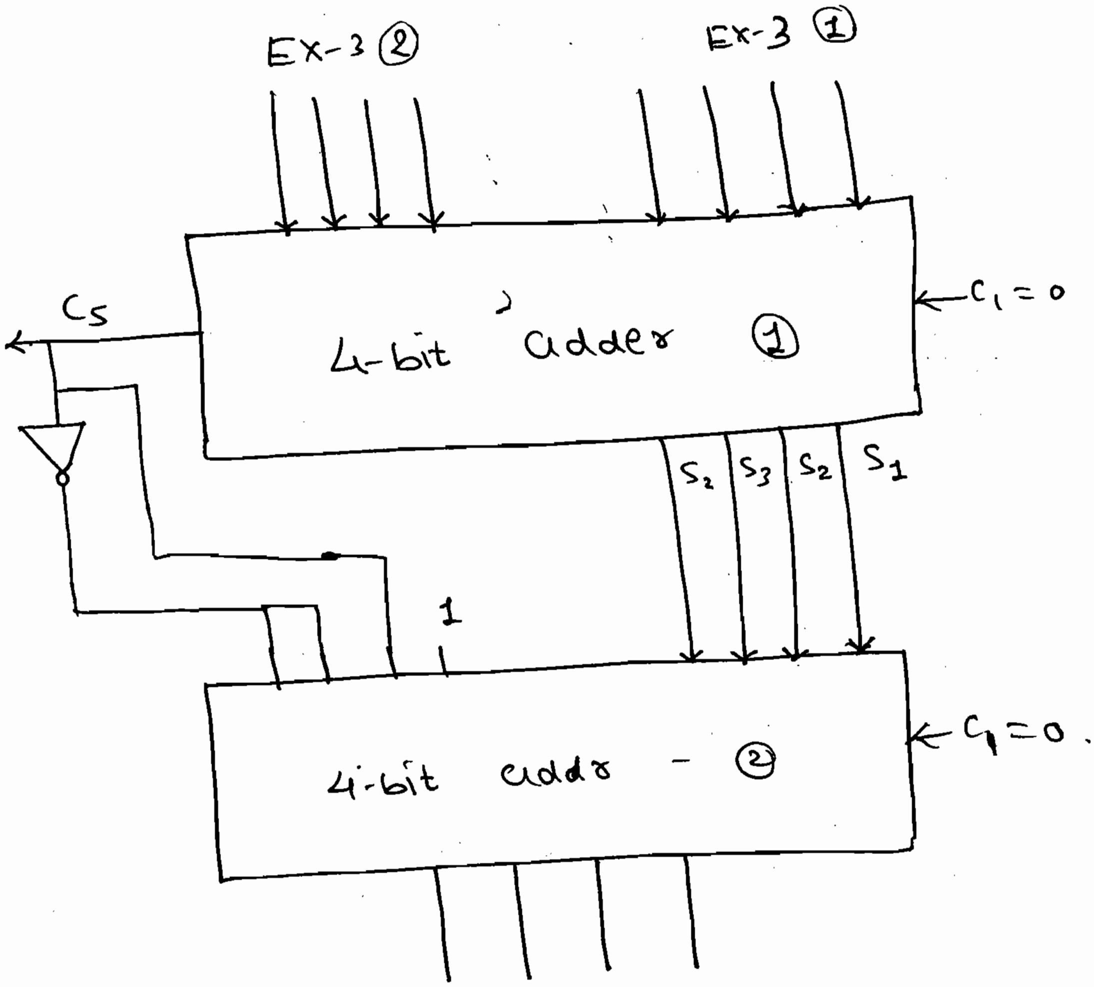
* Ex-3 Adder

(1) If $C_5 = 1 \Rightarrow$ Add $\underline{\underline{\begin{matrix}\bar{C_5} & \bar{C_5} & C_5 & \\ 0 & 0 & 1 & 1\end{matrix}}}$

(2) If $C_5 = 0 \Rightarrow$ Subtract 0011

= Add 2's Comp of 0011

= Add $\underline{\underline{\begin{matrix}\bar{C_5} & C_5 & \bar{C_5} & \\ 1 & 1 & 0 & 1\end{matrix}}}$



* 2-Bit Magnitude Comparator:

→ The truth table of n bit magnitude comparator is not prefered for the design as the no. of rows in the table of 2^{2n}.

n-bit mag comparator ⟶ No. of rows $= 2^{2n}$
$= 2^{2n}$

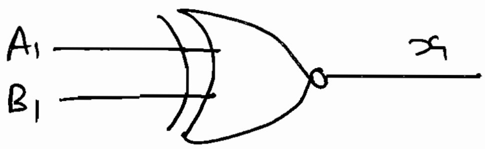
*$\boxed{Q}$
→ How many possible way in 2 bit mag. comparator that $A > B$?

Ans:

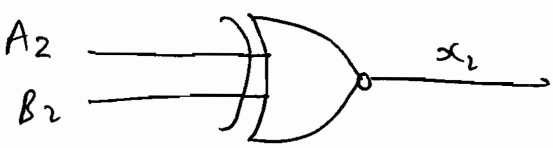
$A_2 A_1$	$B_2 B_1$	
0 0	—	
0 1	0 0	→ 1
1 0	0 0, 0 1	→ 2
1 1	0 0, 0 1, 1 0	→ 3

So, 6 possibility that $A > B$.

*



$x_1 = 1$ if $A_1 = B_1 \Rightarrow x_1 = 1$

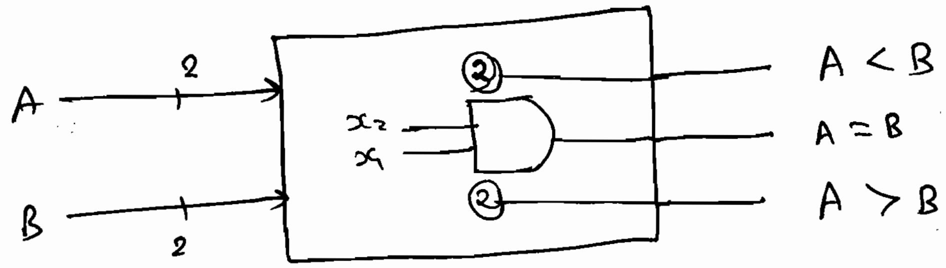


$x_2 = 1$ if $A_2 = B_2 \Rightarrow x_2 = 1$.

① $A = B$ if $A_2 = B_2$ and ~~...~~ $A_1 = B_1$

i.e $A = B$ if $x_2 = 1$ and $x_1 = 1$.

i.e $A = B$ if $x_1 \cdot x_2 = 1$.



$A = A_2 A_1$.
$B = B_2 B_1$.

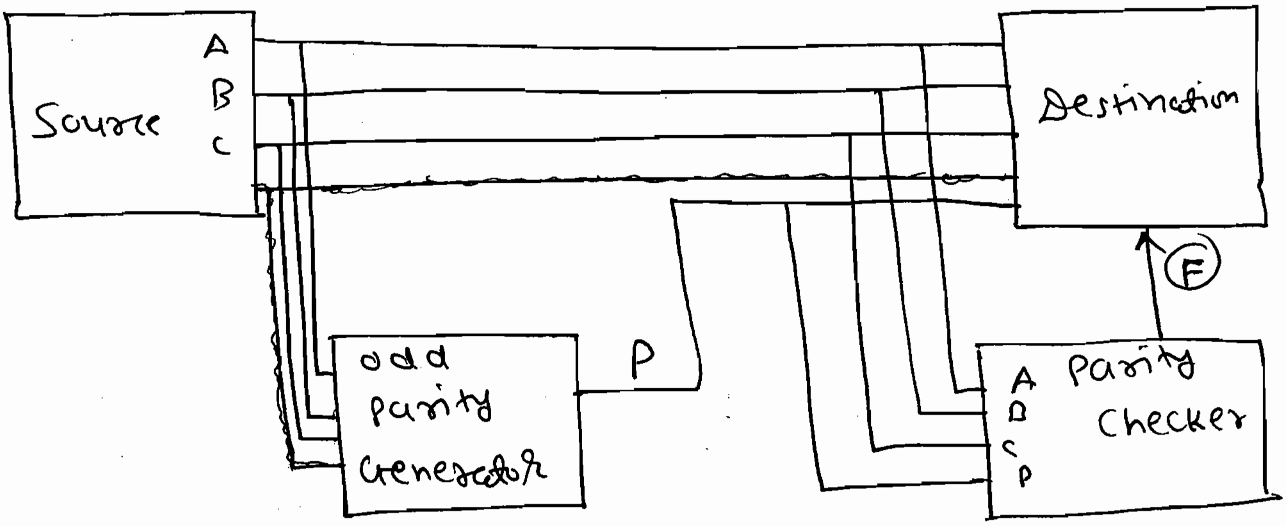
② $A > B$ if $A_2 > B_2$ (or) $A_2 = B_2$ and $A_1 > B_1$.

i.e. $A > B$ if $\boxed{A_2 \bar{B_2} + x_2 \cdot A_1 \bar{B_1} = 1}$ — ②

③ $A < B$ if $A_2 < B_2$ (or) $A_2 = B_2$ and $A_1 < B_1$.

i.e $A < B$ if $\boxed{\bar{A_2} B_2 + x_2 \cdot \bar{A_1} B_1 = 1}$ — ③.

* Odd parity ~~Gen~~ Generator and Parity Checker



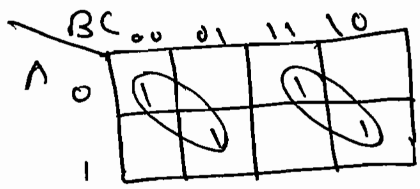
P is choosen so that
A, B, C, P → odd parity.

F = 1 if even parity
F = 0 if odd parity
occur for A, B, C, P.

→ odd parity generator:

A	B	C	P
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

$P(A, B, C) = \Sigma m(0, 3, 5, 6)$



$P = A \oplus B \odot C$

(or)

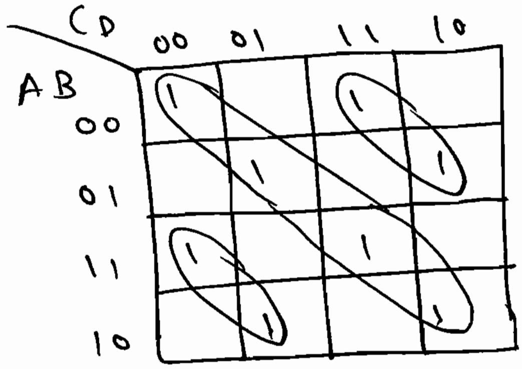
$P = A \odot B \oplus C.$

Parity Checker.

A	B	C	P	F
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
Φ	0	0	0	0
Φ	0	0	1	1
Φ	0	1	0	1
Φ	0	1	1	0
Φ	1	0	0	1
Φ	1	0	1	0
Φ	1	1	0	0
Φ	1	1	1	1

$F = \Sigma m(A, B, C, P)$

$= \Sigma m(0, 3, 5, 6, 9, 10, 12, 15).$



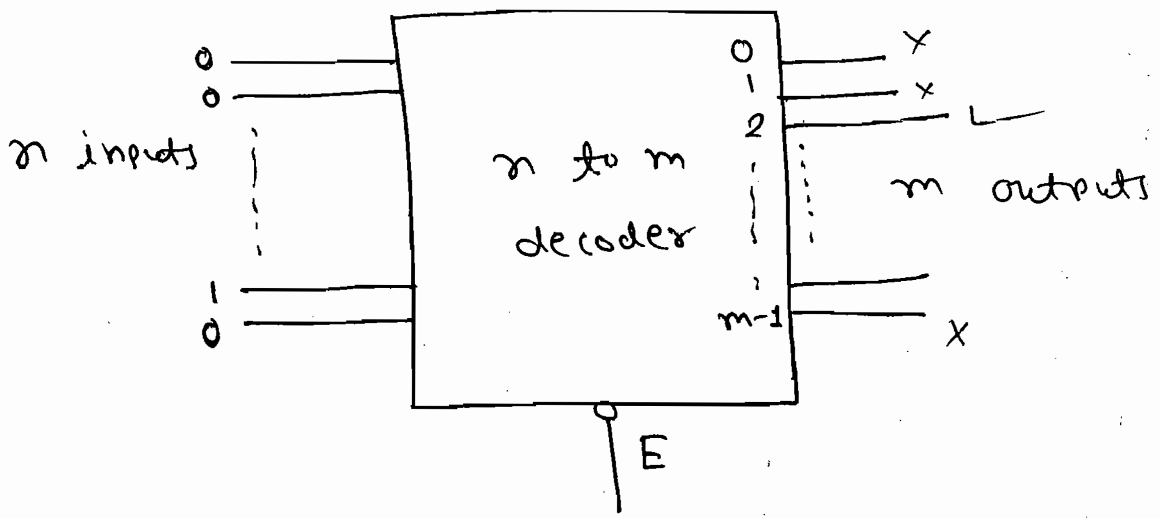
* All minterms have Even no of 0's.

$\Rightarrow$ $\boxed{F = A \odot B \odot C \odot P.}$

* Decoder:

→ It converts the binary information on i/p lines to one of many O/P lines.

→ n to m decoder = (1 out of m decoder).

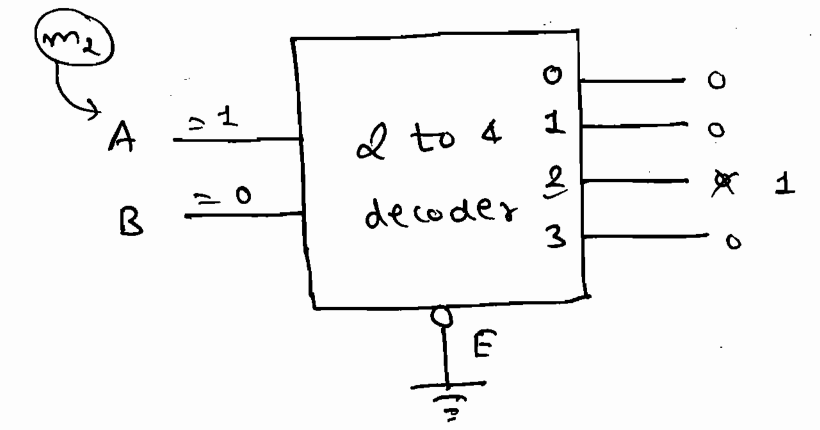


E=0 ⇒ Decoder is enabled

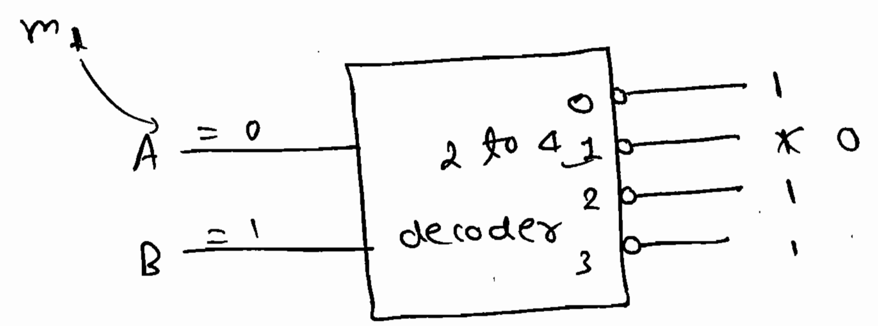
E=1 ⇒ Decoder is desabled

$$\boxed{m \le 2^n}$$

* 2 to 4 decoder (Active High o/p).

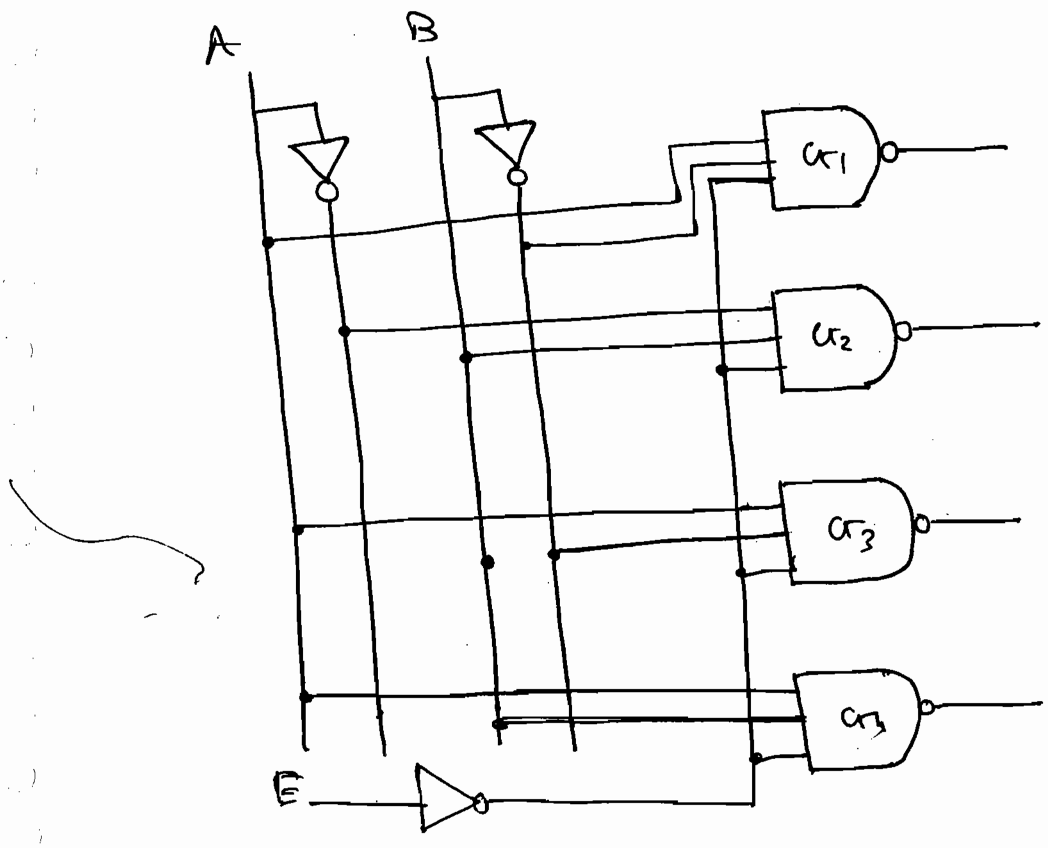


* 2 to 4 decoder (Active Low o/p).



Most widely used in practice because it generate less noise compare to Active High Decoder.

⇒ Internal circuits:



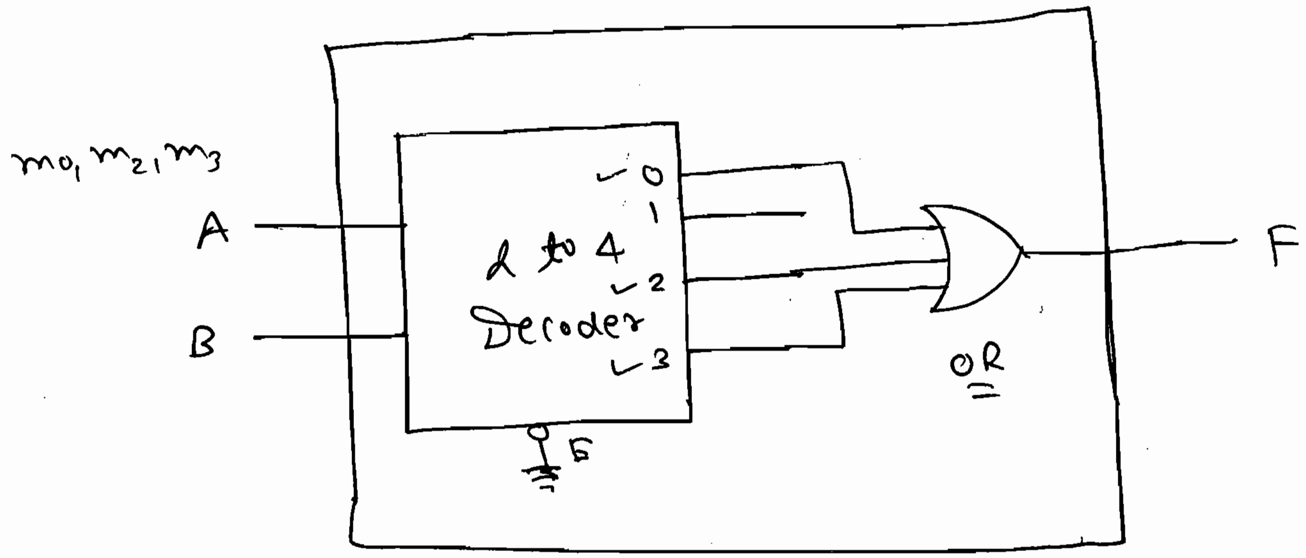
* Decoders
  ⇒ AND, NAND gates.

Ex-1 Implement $F(A,B) = \Sigma m(0,2,3)$. using a decoder (a) with active high o/p.s
(b) with active Low o/ps.

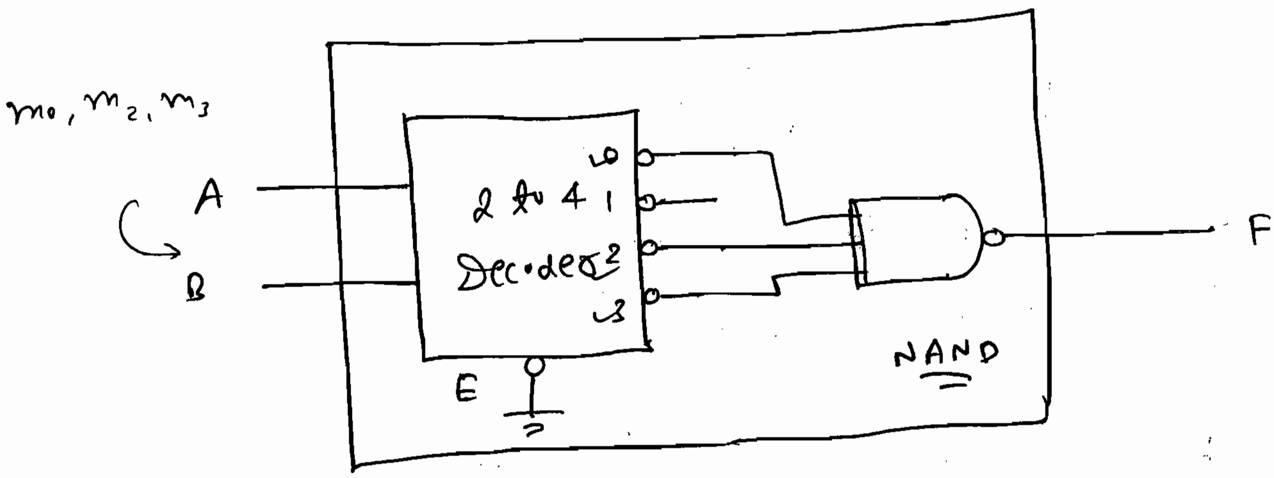
Ans: (a) With active high o/p.

A	B	F
0	0	1
0	1	0
1	0	1
1	1	1

m_0
m_2
m_3



(b) With active Low o/p.

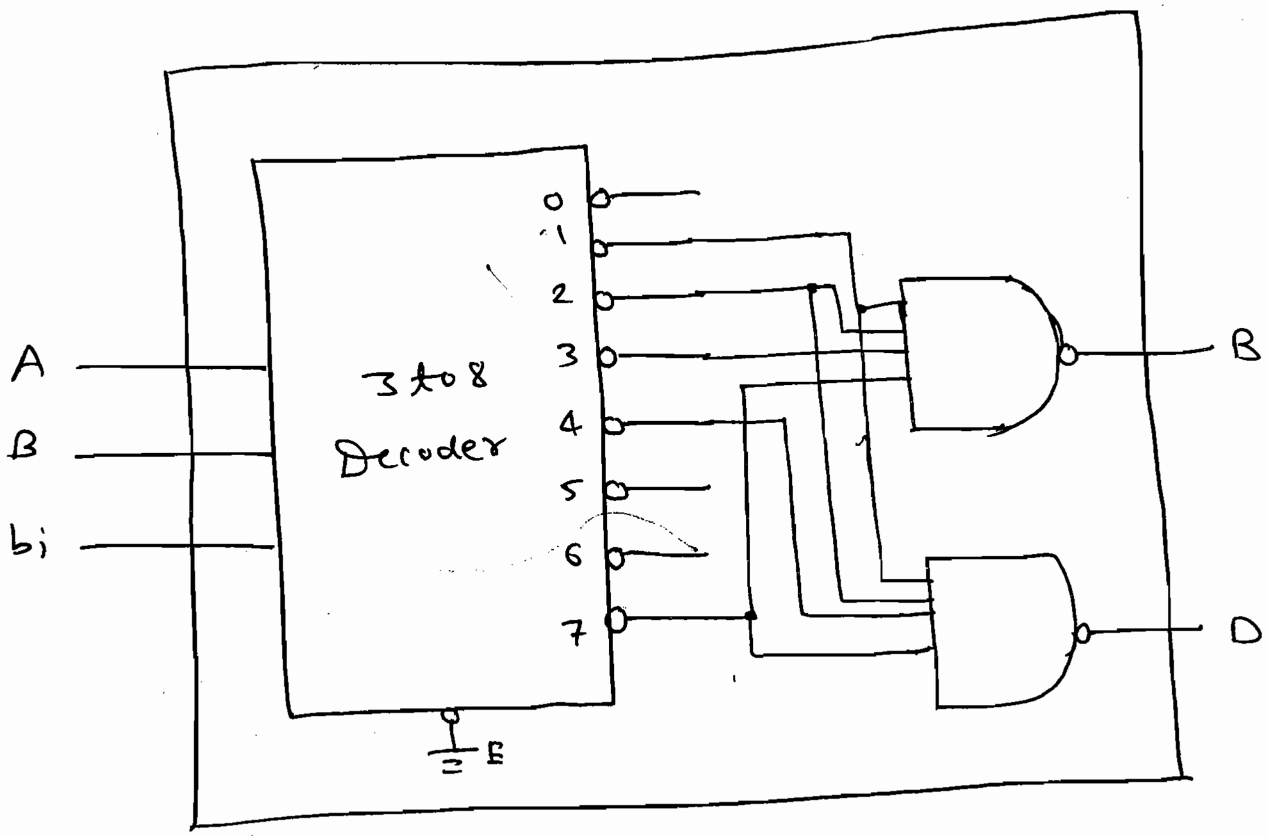


Ex-2 Implement a full Subtractor using a decoder with active Low outputs.

Ans:

$D(A, B, b_i) = \Sigma m(1, 2, 4, 7)$.

$B(A, B, b_i) = \Sigma m(1, 2, 3, 7)$.



Ex 3 How many 2 to 4 decoder required to construct 1 out of 32 decoder.

Ans: <u>Short cut</u>:

$$\frac{\text{Required No. of O/P}}{\text{Given no. of o/p}}$$

No. of Decoders

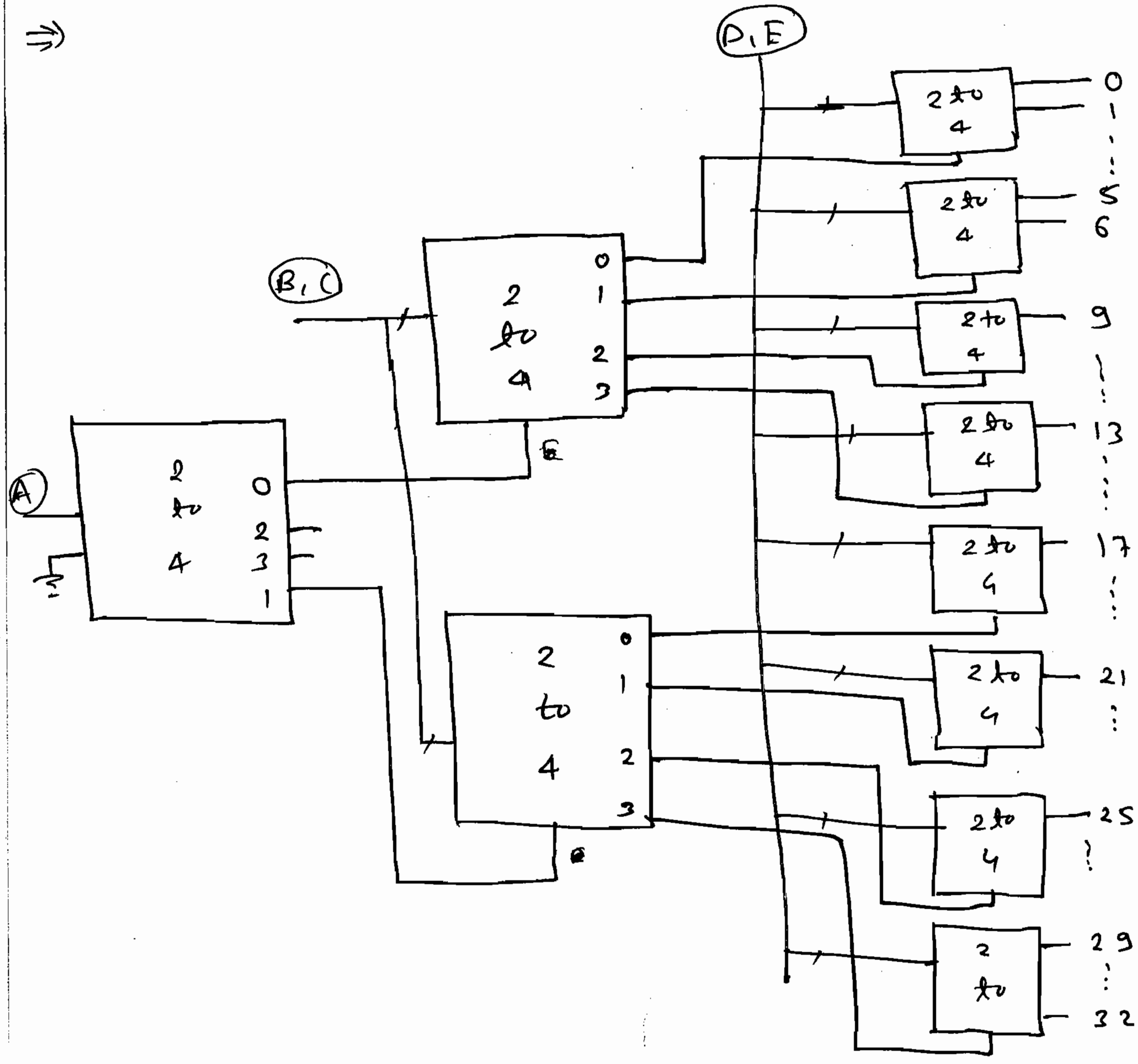
$\therefore \frac{32}{4} = 8$ 8

$\frac{8}{4} = 2$ $+ \; 2$ } 3 levels.

$\frac{2}{4} = 0.5 \approx 1$ $+ \; 1$

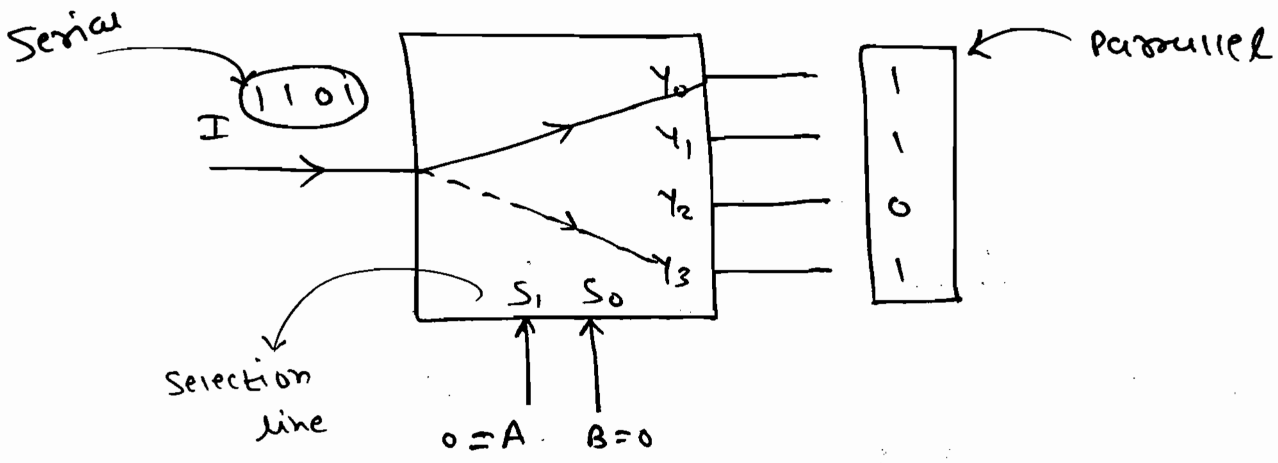
11

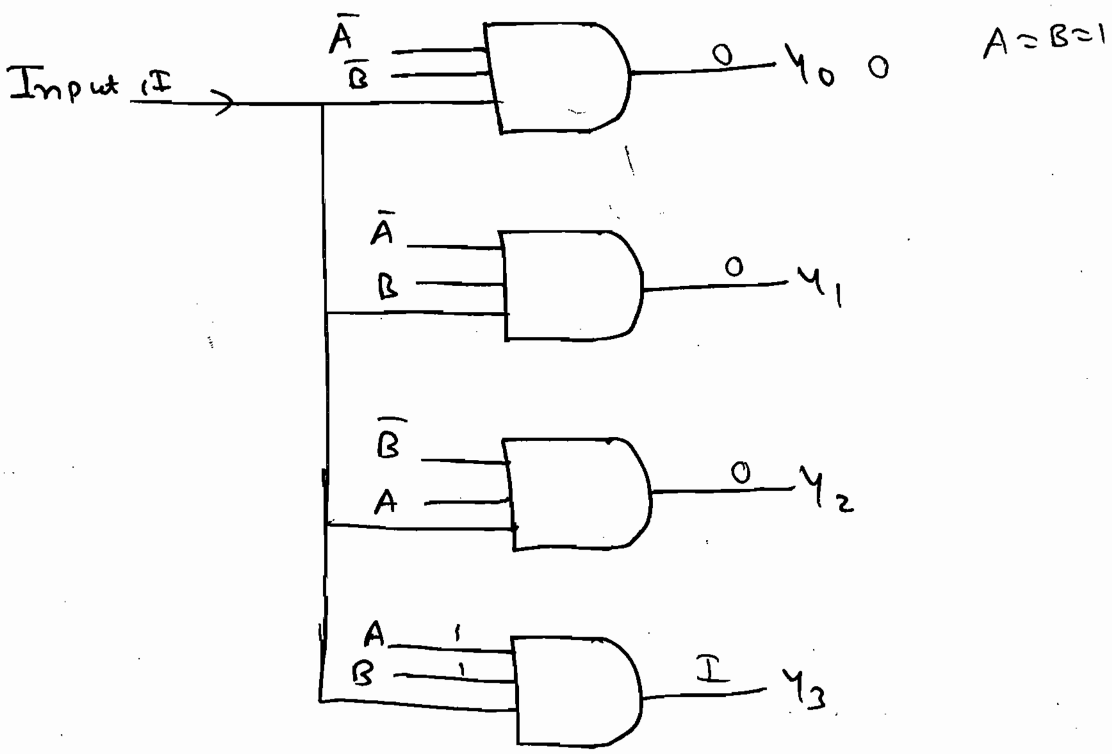
$\Rightarrow$



* Demultiplexer (one to many ckts, serial to parallel converter).

1 : 4 Demux





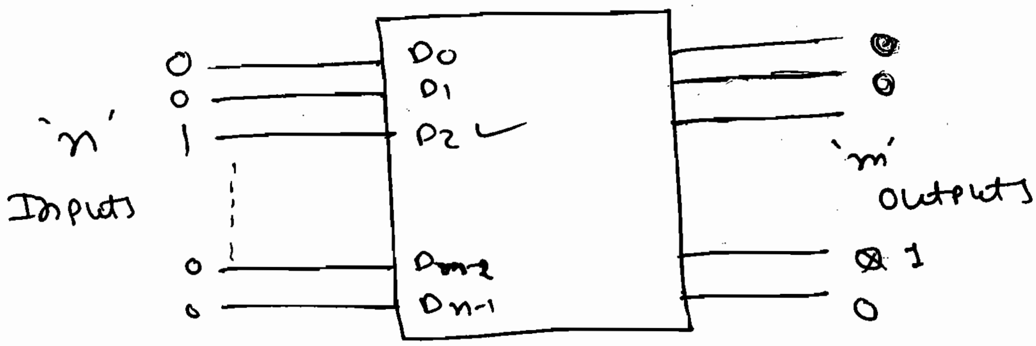
→ A Demux is similar to a decoder.

→ A 1:4 Demux is converted to a 2 to 4 Decoder by making two changes.

(1) Selection of Demux are converted to the i/ps of 2 to 4 decoder.

(2) The demux input I is treated as active high enable of 2 to 4 decoder.

* Encoder:



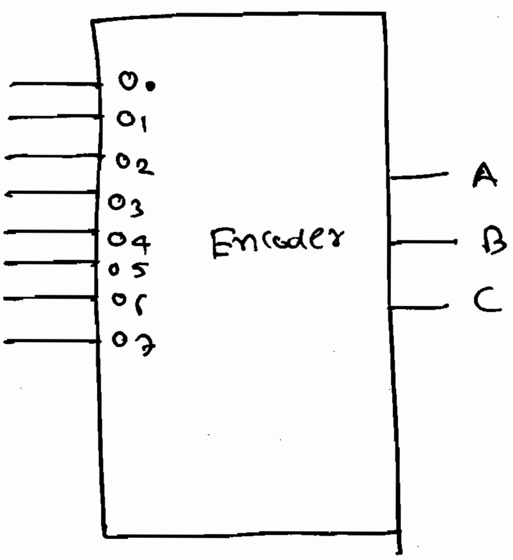
Input selected $\longrightarrow$ Output code

If $D_1 = 1$ $\longrightarrow$ 000.... 01

If $D_2 = 1$ $\longrightarrow$ 000.... 10.

If $D_1 = 1, D_2 = 1$ $\longrightarrow$ * Coding can not be performed.

Ex-1 Design octal to Binary encoder.



O_7	O_6	O_5	O_4	O_3	O_2	O_1	O_0	A	B	C
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	0
0	0	1	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	0	1	1	0
1	0	0	0	0	0	0	0	1	1	1

$A = O_4 + O_5 + O_6 + O_7.$

$B = O_2 + O_3 + O_6 + O_7.$

$C = O_1 + O_3 + O_5 + O_7.$

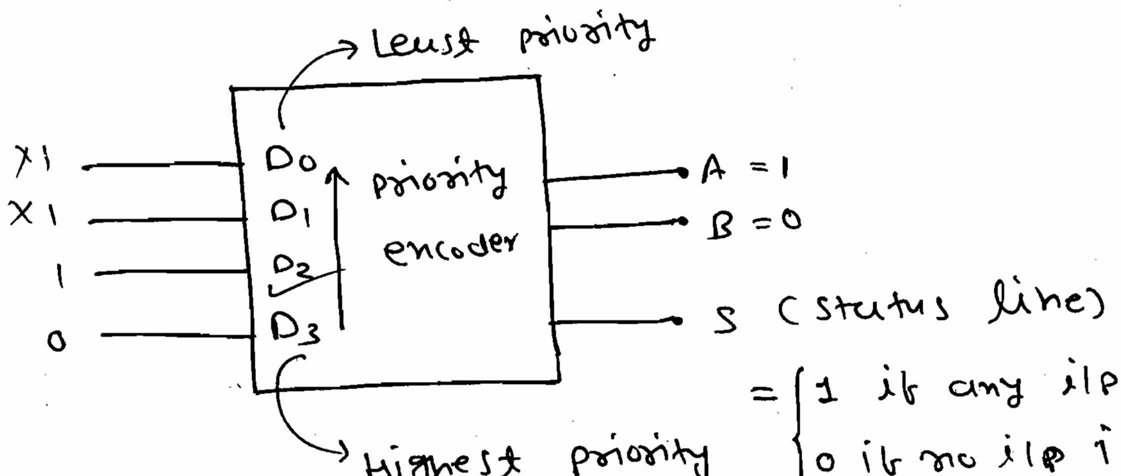
NOTE:

→ The encoder follows the $\boxed{\text{OR Logic}}$.

→ The limitation of encoder is it can't performed the coding if more than 1 input is active simultaneously. To overcome this, we use priority encoder.

* Priority Encoder:

⇒



$= \begin{cases} 1 & \text{if any i/p is applied.} \\ 0 & \text{if no i/p is applied.} \end{cases}$

⇒

	D_3	D_2	D_1	D_0	A	B	S
m_0 ←	0	0	0	0	0	0	0
m_1 ←	0	0	0	1	0	0	1
m_2, m_3 ←	0	0	1	X	0	1	1
m_4 to m_7 ←	0	1	X	X	1	0	1
m_8 to m_{15} ←	1	X	X	X	1	1	1

(b) Priority encoder Table.

→ $B(D_3, D_2, D_1, D_0) = \Sigma m(2, 3, 8, 9, 10, \ldots, 15).$

$D_3D_2 \backslash D_1D_0$	00	01	11	10
00			1	1
01				
11	1	1	1	1
10	1	1	1	1

$B = D_3 + \overline{D_2} D_1$

→ $A(D_3, D_2, D_1, D_0) = \Sigma m(4, 5, 6, 7, \ldots, 15).$

$D_3D_2 \backslash D_1D_0$	00	01	11	10
00				
01	1	1	1	1
11	1	1	1	1
10	1	1	1	1

$A = D_2 + D_3$

→ $S = \Sigma m(1, 2, 3, \ldots, 15).$

$D_3D_2 \backslash D_1D_0$	00	01	11	10
00	0	1	1	1
01	1	1	1	1
11	1	1	1	1
10	1	1	1	1

$\therefore S = D_3 + D_2 + D_1 + D_0$

$S = D_3 + D_2 + D_1 + D_0$

NOTE:

74LS138 ⇒ { 3 to 8 decoder (Active low outputs).

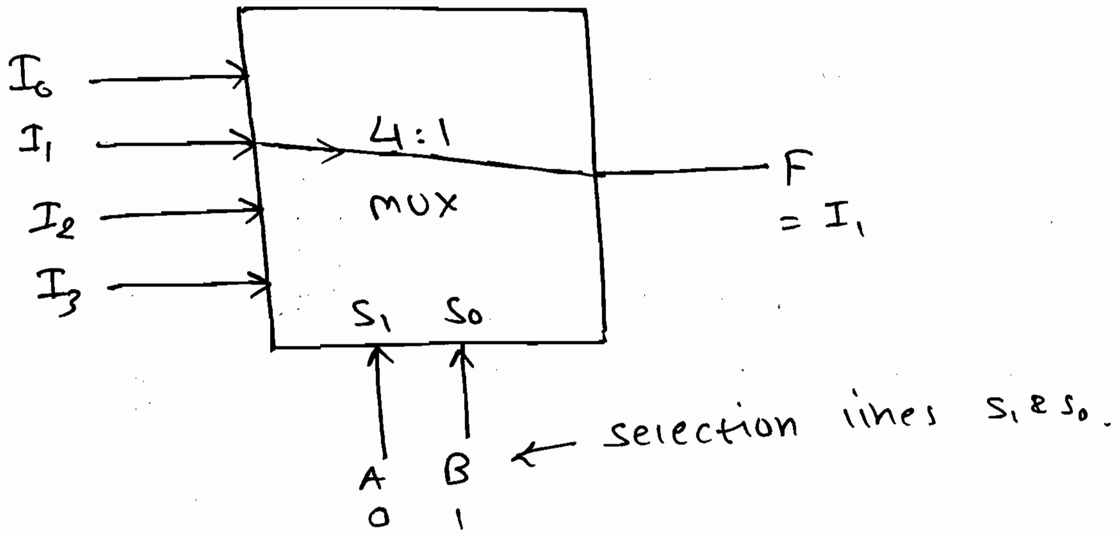
74LS148 ⇒ { 8-input, 3-output priority Encoder.

* Multiplexer:

(Many to one circuit, parallel to serial converter).

*

4 : 1 MUX.



∴

A	B	F
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

⇒ $\boxed{4:1 \text{ MUX} :\rightarrow \ F = \bar{A}\bar{B}I_0 + \bar{A}BI_1 + A\bar{B}I_2 + ABI_3.}$

$$\therefore \boxed{F = m_0 I_0 + m_1 I_1 + m_2 I_2 + m_3 I_3.}$$

* E.g.: Given $I_0 = I_1 = 1$; $I_2 = I_3 = 0$.

$$\therefore F = m_0 \cdot 1 + m_1 \cdot 1 + 0 + 0.$$

$$\therefore F = m_0 + m_1$$

$$\therefore \boxed{F = \sum m(0,1).}$$

Ex-1 Implement the Sum o/p of half Adder using 4:1 MUX.

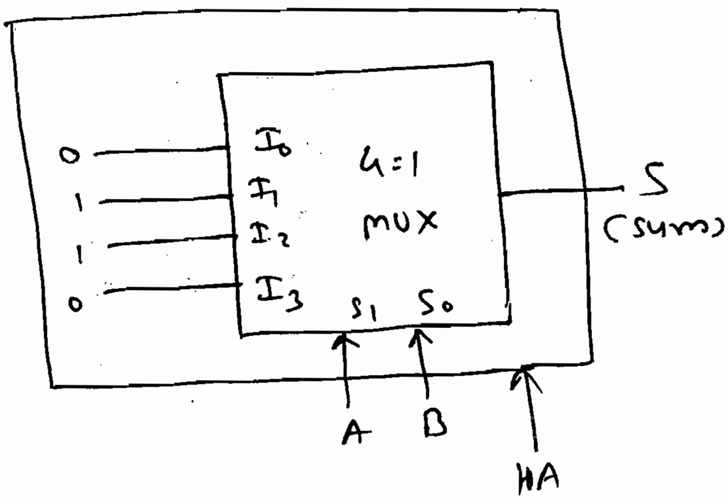
Ans:

$S(A,B) = m\sum(1,2)$.

$S(A,B) = m_1 + m_2$

Choose $I_1 = I_2 = 1$ &

$I_0 = I_3 = 0$



Ex-2 Which of the following is implemented by the following mux.

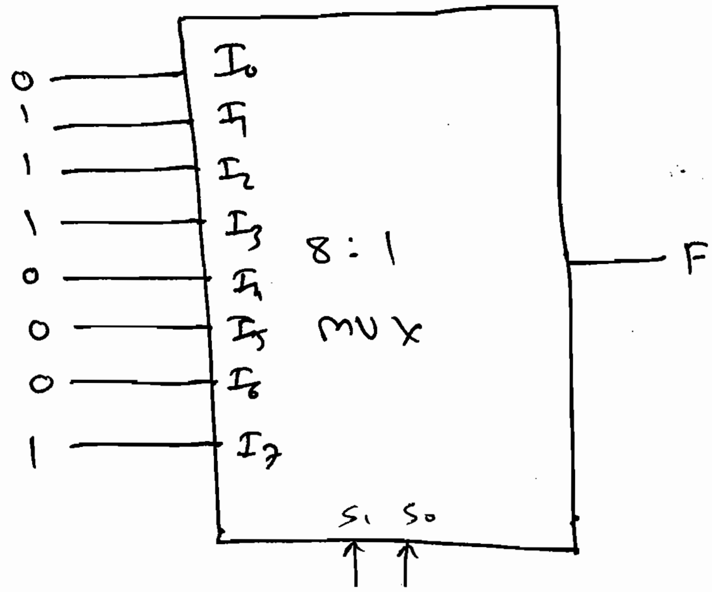
(a) Sum output of Full adder.

(b) Carry output of Full adder.

(c) Difference output of Full Subtractor.

✓(d) Borrow output of Full Subtractor.

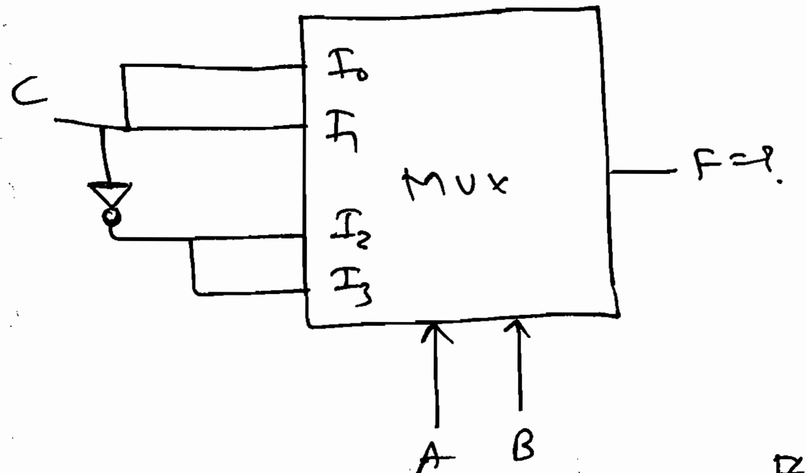
(b) ~~Sum o/p of full adder.~~



$$F = \Sigma m(1, 2, 3, 7)$$

Ex-2 What are the logic gates represented by the following mux circuits.

(a)



$$F = m_0 C + m_1 C + m_2 \bar{C} + m_2 \bar{C}$$

$$F = (\bar{A}\bar{B} + \bar{A}B) C + (A\bar{B} + AB)\bar{C}$$

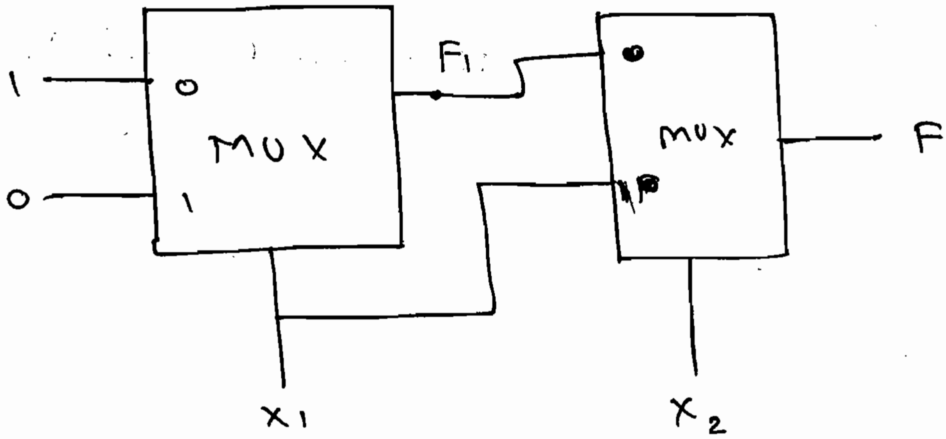
$$F = (\bar{A}\bar{B} + \bar{A}B)\, C + (A\bar{B} + AB)\,\bar{C}$$

$$= \bar{A}C + A\bar{C}$$

$$F = A \oplus C$$

$\therefore$ Ex-OR gate

(b)



$\therefore \quad F_1 = m_0 = \overline{x_1}$

$\therefore \quad F_1 = \overline{x_1}$

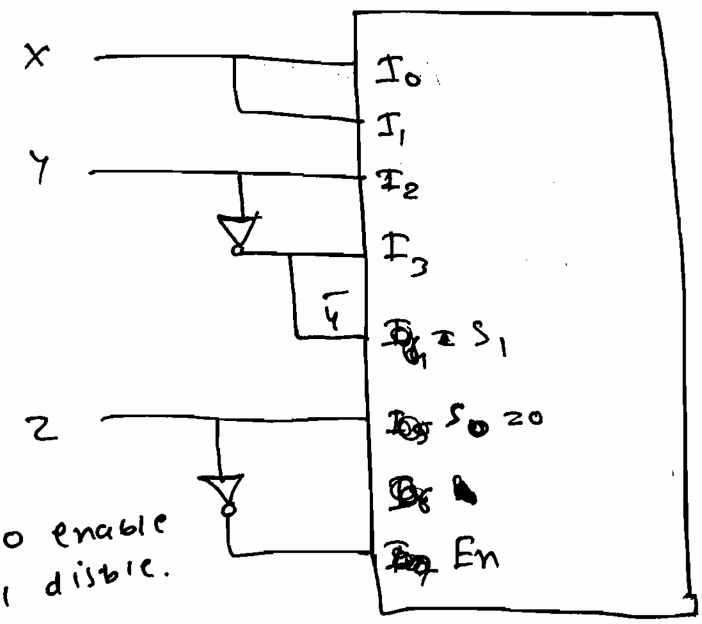
$\therefore \quad F = \overline{F_1}\,\overline{x_2} + x_2 x_1$

$= \overline{\overline{x_1}}\cdot\overline{x_2} + x_2 x_1$

$\therefore \quad \boxed{F = x_1 \odot x_2}$

Ex-4 Determine the output of the following MUX.

Ans.



$F = (m_0 + m_1)\,x + m_2 y + m_3 \bar{y}$

$= (\bar{S_1}\bar{S_0} + \bar{S_1} S_0)\,x + S_1 \bar{S_0}\, y + S_1 S_0 \bar{y}$

$= \bar{S_1}\,x + S_1 \bar{S_0}\, y + S_1 S_0 \bar{y}$

$= \bar{S_1}\,x + S_1 y + 0$

$= xy + 0$

$= xy$

$\Rightarrow \boxed{F = x \cdot y \cdot \bar{z}}$

$\Rightarrow$ $z = 0 \rightarrow$ enable.

$z = 1 \rightarrow$ disable.

Ex-5 How many 4:1 mux required to construct 128 :1 mux.

Ans: $\dfrac{\text{Required No. of Inputs}}{\text{Given No. of I/p}}$

No. of mux

$= \dfrac{128}{4} = 32 \longrightarrow 32$

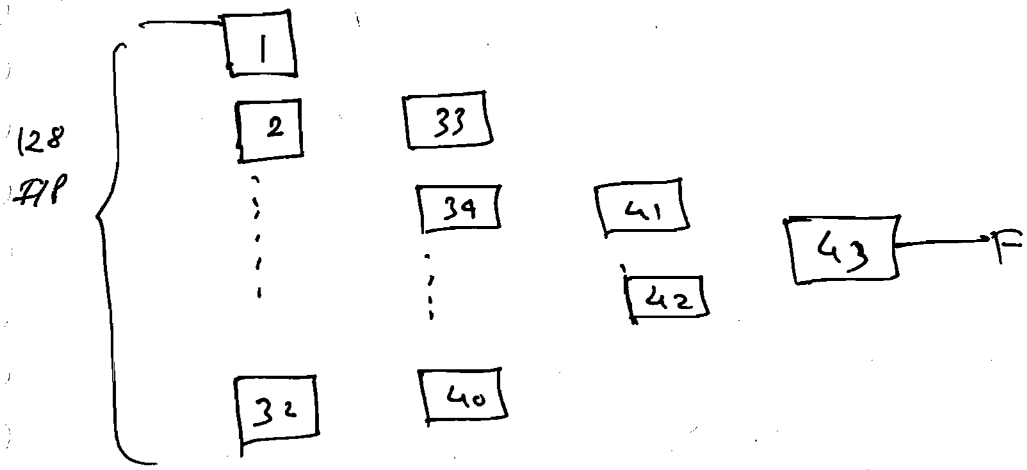
$+$

$\therefore \dfrac{32}{4} = 8 \longrightarrow 8$

$\dfrac{8}{4} = 2 \longrightarrow +2$

$\dfrac{2}{4} = 0.5 \longrightarrow +1$

$43.$



(b) 2:1 MUX $\longrightarrow$ $2^n:1$ MUX.

$\longrightarrow$

No. of 2:1 mux

2:1 MUX $\longrightarrow$ 4:1 MUX. $\longrightarrow$ 3

$\longrightarrow$ 8:1 MUX $\longrightarrow$ 7

$\longrightarrow$ 16:1 MUX $\longrightarrow$ 15.

⋮

$\longrightarrow$ $2^n:1$ MUX $\longrightarrow$ $2^n - 1$.

So, $\boxed{2^n - 1}$ 2:1 MUX required for $2^n:1$ MUX.

Ex-5 How many 2 : 1 MUX are req. to realize
ⓐ AND gate
ⓑ OR gate
ⓒ EX-OR gate.

Ans:

$F = \bar{A} I_0 + A I_1$

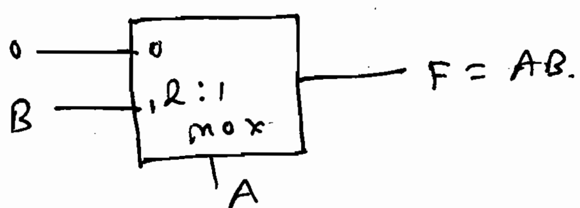
$F = AB$

$F = AB + 0 \cdot \bar{A}$

$F = 0 \cdot \bar{A} + B \cdot A.$

$F = \bar{A} I_0 + A I_1$

So, $I_0 = 0$, $I_1 = B$.



(b) OR gate:

$\therefore F = \bar{A} I_0 + A I_1$

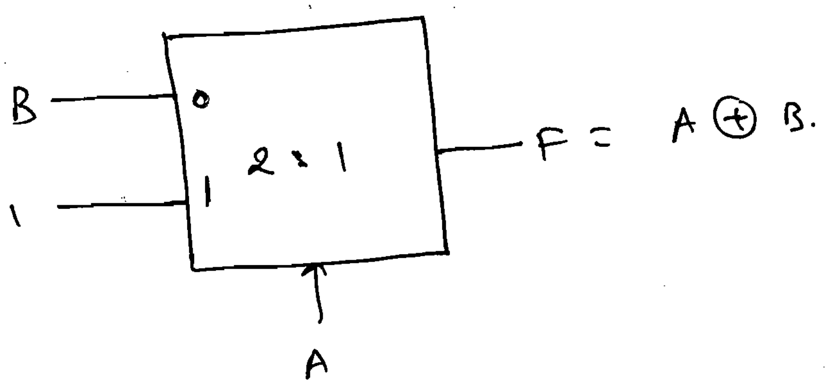
$F = A + B.$

$= A + \bar{A} B$

$F = \bar{A} B + A \cdot 1.$

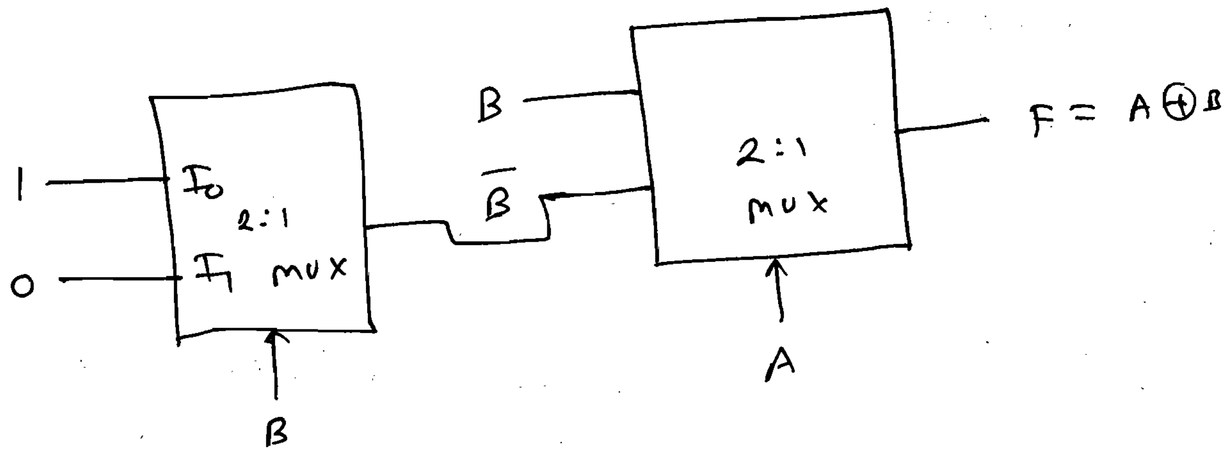
$\therefore F = \bar{A} I_0 + A \cdot I_1$

$\therefore$ So, Choose $I_0 = B$, $I_1 = 1$.



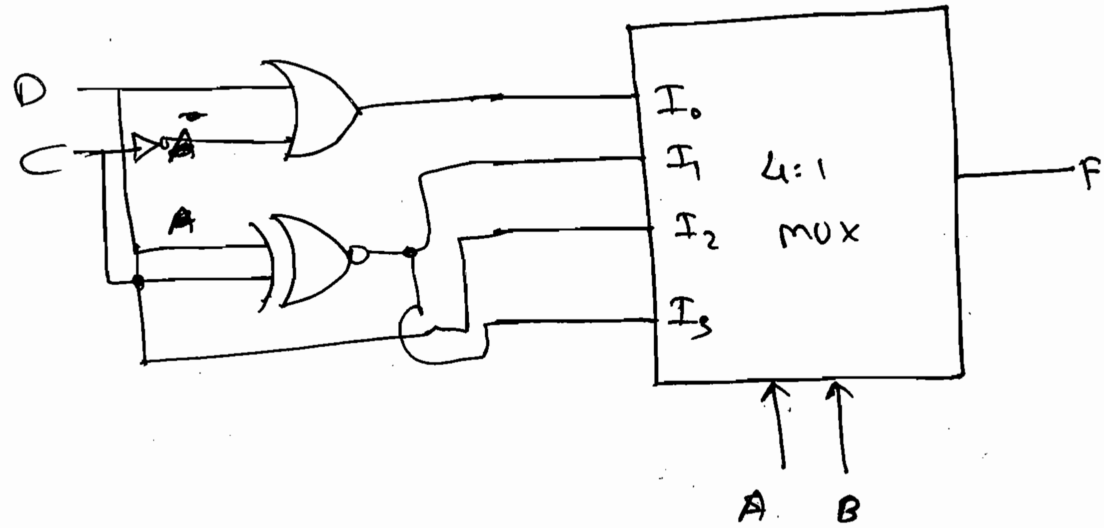
(c) Ex-OR gate:

$\rightarrow \quad F = \bar{A} \underset{\underset{I_0}{\uparrow}}{B} + A \underset{\underset{I_1}{\uparrow}}{\bar{B}} = A \oplus B.$



Ex-6 Implement $F(A, B, C, D) = \Sigma m(0, 1, 3, 5, 6, 10, 11, 13, 14)$ using 4:1 mux.

Ans:



Method - 1

	AB=	00	01	10	11
		I_0	I_1	I_2	I_3
00	$\bar{C}\bar{D}$	0	4	8	12
01	$\bar{C}D$	1	5	9	13
10	$C\bar{D}$	2	6	10	14
11	CD	3	7	11	15
		$\bar{C}\bar{D} + \bar{C}D + CD = \bar{C} + D$	$C \oplus D$	C	$C \oplus D$.

Method :- 2

If A, B are selection line

AB \ CD	00	01	11	10
00	1	1	1	
01		1		1
10		1	a	1
11			1	1

$\rightarrow I_0 = \bar{C}\bar{D} + \bar{C}D + CD = \bar{C} + D$

$\rightarrow I_1 = \bar{C}D + C\bar{D} = C \oplus D.$

$\rightarrow I_2 = \bar{C}D + C\bar{D} = C \oplus D$

$\rightarrow I_3 = CD + C\bar{D} = C$

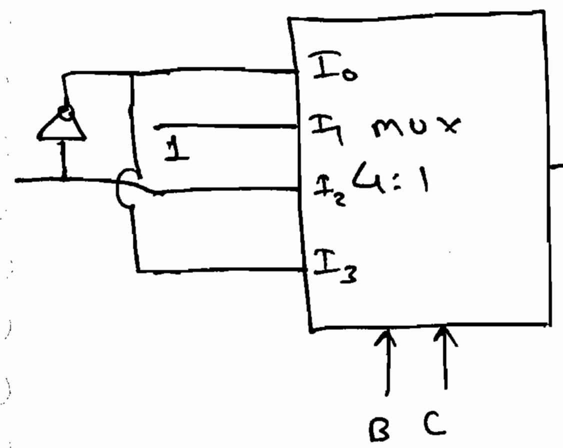
→ **NOTE:** If C & D are selection lines

AB \ CD	00	01	11	10
00	1	1	1	
01		1		1
11		1		1
10			1	1

$I_0 = \bar{A}\bar{B}$ $\quad I_1 = \bar{A}+B$ $\quad I_3 = \bar{B}$ $\quad I_2 = A+B.$

Ex-1 $F(A,B,C) = \Sigma m(0,1,3,5,6)$ using 4:1 MUX.

Ans:



Method-I:

	I_0	I_1	I_2	I_3
$\bar{A}$	0	1	2	3
A	4	5	6	7
	$\bar{A}$	1	A	$\bar{A}$

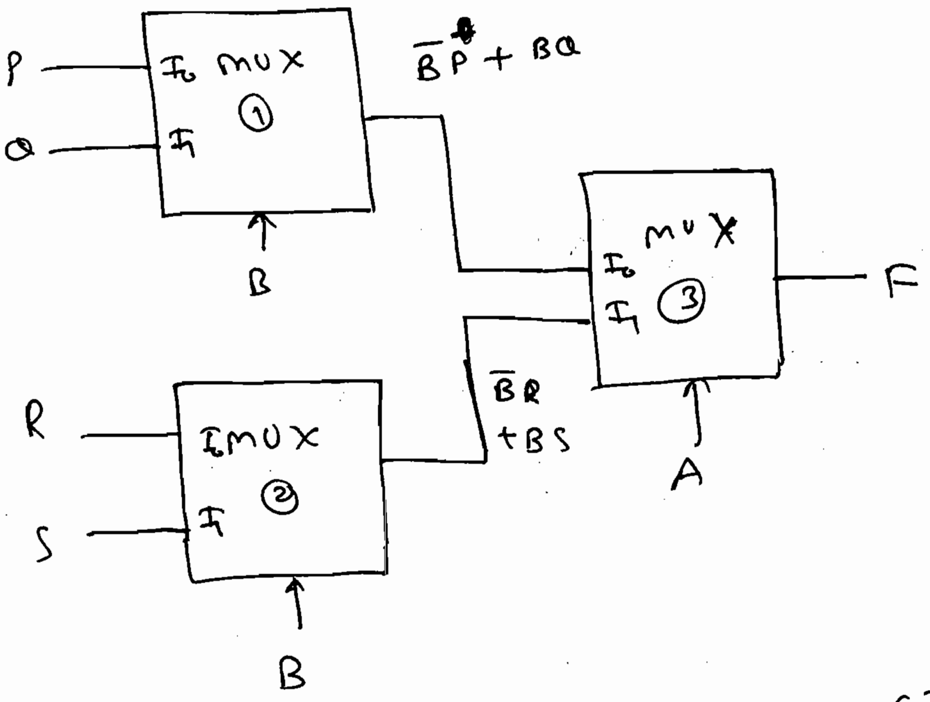
Method-II (K-MAP)

A \ BC	00	01	11	10
0	1	1	1	
1		1		1

$I_0 = \bar{A}$ $\quad I_1 = 1$ $\quad I_3 = \bar{A}$ $\quad I_2 = A$

Ex-2: In the following Mux tree, determine the values of $P, Q, R, S = ?$

Where, $F(A,B,C) = \Sigma m(1,2,4,5,6)$.



$$\therefore \quad F = \bar{A}(\bar{B}P + BQ) + A(\bar{B}R + BS).$$

$$\therefore \quad F = \bar{A}\bar{B}P + \bar{A}BQ + A\bar{B}R + ABS.$$

C \ AB	00	01	11	10
0	(0)	(2) 1	(6) 1	(4) 1
1	(1) 1	(3)	(7)	(5) 1

$I_0 = C$ → P, $I_1 = \bar{C}$ → Q, $I_3 = \bar{C}$ → R, $I_2 = 1$ → S

$$F = \bar{A}\bar{B}C + \bar{A}B\bar{C} + A\bar{B}\bar{C} + A\bar{B}C + AB\bar{C}$$

$$\therefore F = \bar{A}\bar{B}C + \bar{A}B\bar{C} + A$$

Ex-2 Using $n:1$ MUX we can implement all '$\log_2 N$' variable function and some of "$\log_2 N+1$" variable functions [T / F].

Ans: let, 4:1 MUX

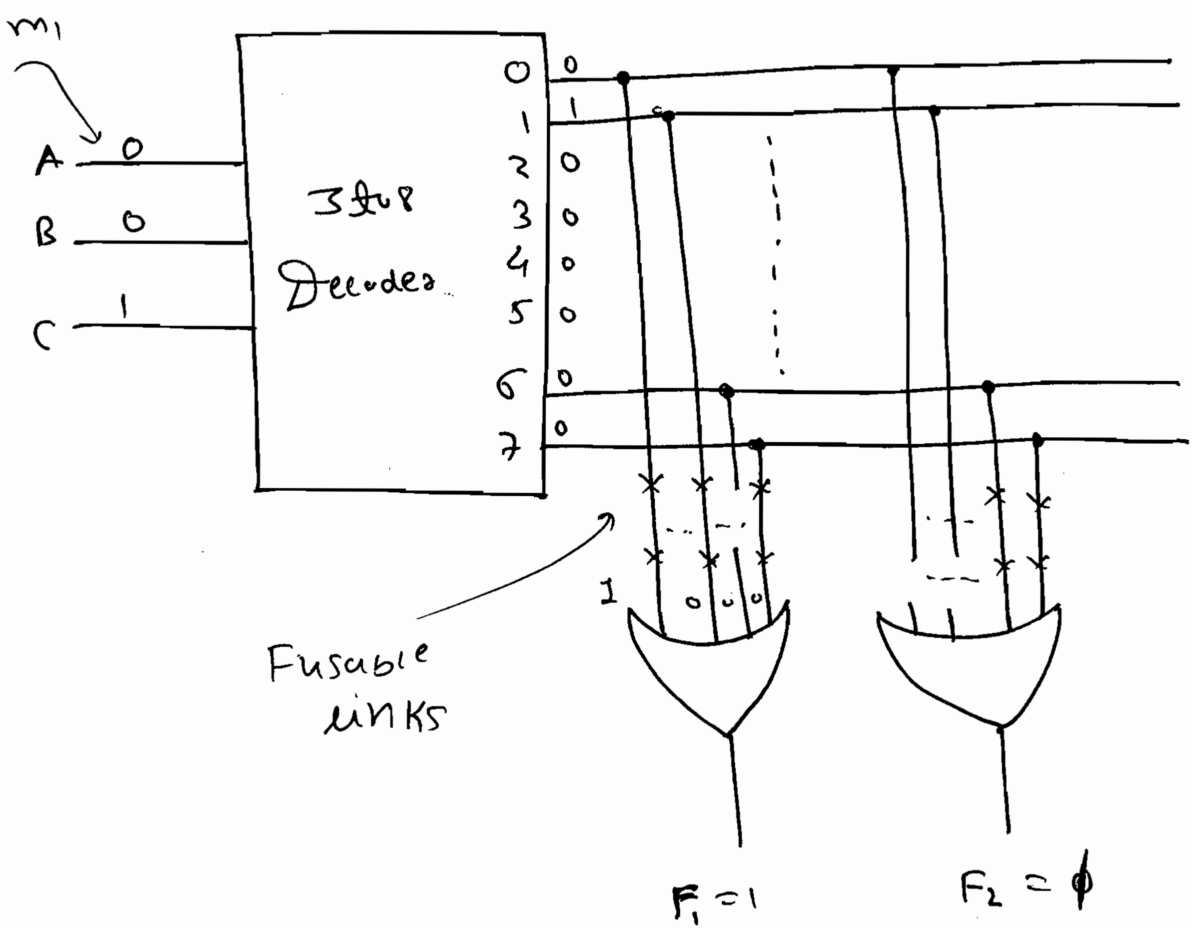
so, $\log_2 4 = 2$

→ True : because the other functions required external logic gates along with MUX.

* ROM (Read only memory).

⇒ Decoder (Fixed AND gates) + Prog. OR gates (encoder) } = ROM.

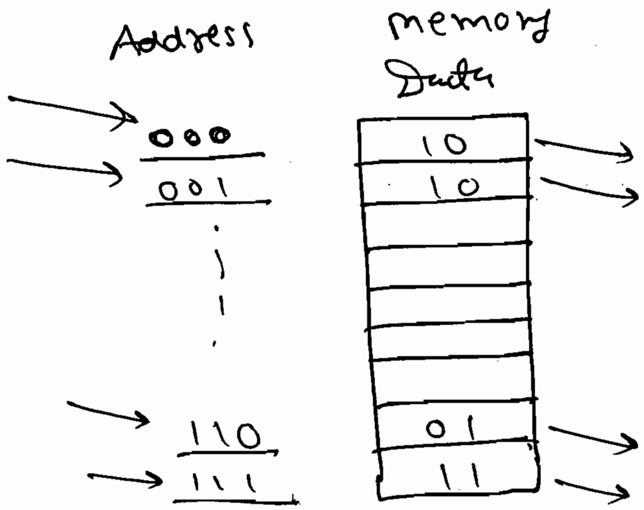
→ ROM is a combinational circuits and we can use it to implement sum of minterms expression as shown in the following example:



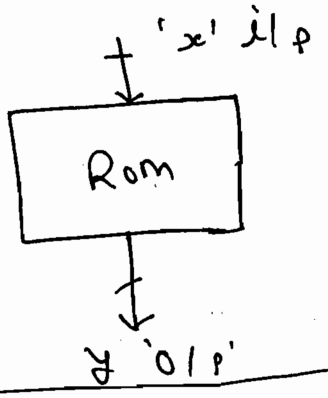
$F_1(A, B, C) = \Sigma m(0, 1, 7)$. size $= 2^3 \times 2 = 16$.

$F_2(A, B, C) = \Sigma m(6, 7)$.

⇒ $\boxed{\text{Read only memory}}$



⇒



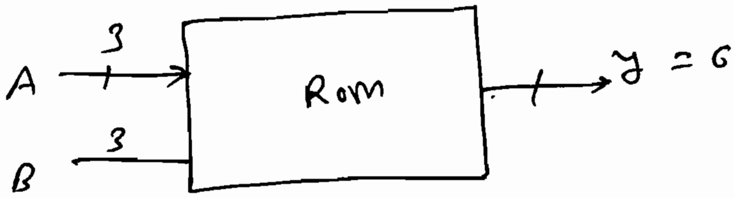
∴ $\boxed{\text{Rom size} = 2^x \times y.}$

→ Rom size indicates that how many bits can be stored.

Ex-1 Determine the size of the Rom for the following functions.

① Rom of 3 bit binary multiplier.

②



$y = 3+3 = 6$

$y = 6$ $\qquad \overset{111}{7}_{10} * \overset{111}{7}_{10} = 49_{10}$

$$2^y > 49$$

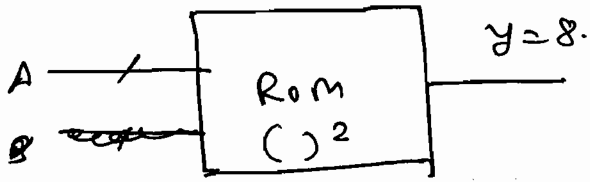
$$\boxed{y = 6}$$

$\therefore$ Rom size = $2^x \times y = 2^6 \times 6 = 384.$

(2) Rom as 4-Bit squarer.

Ans:

$X = 8$



~~64~~

1111

1111

~~64~~

$\cancel{64 \times 64}$

$\cancel{2^4 \times 2^4}$

$\cancel{y = 8}$

$\cancel{x = 4}$. $\qquad \cancel{2^4 \times 8 = 16 \times 8 = 128}$.

$15 \times 15 = 225_{10}$

$\therefore$

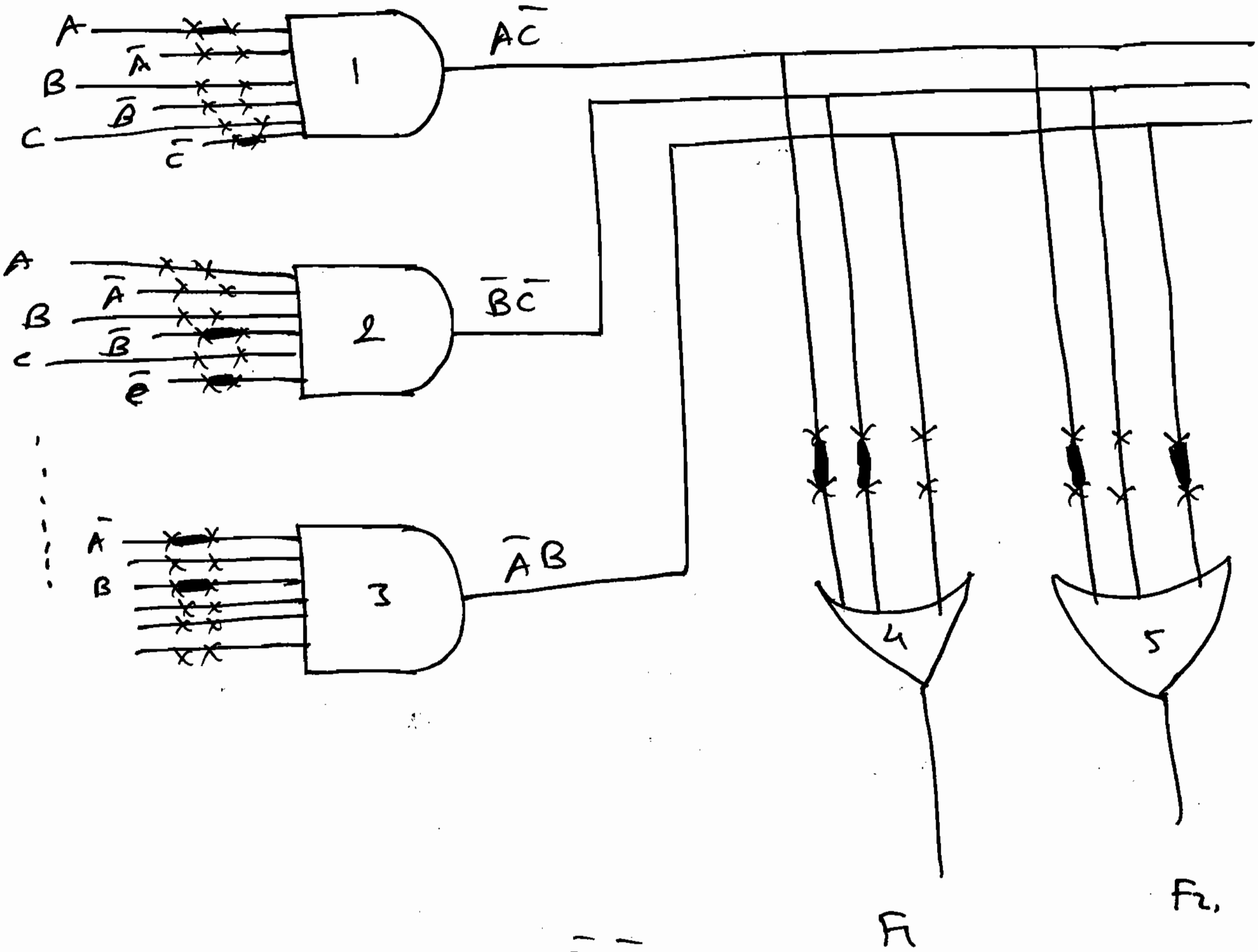
$2^y > 225$

$\boxed{y = 8}$, $x = 4$.

$\therefore$ size = $2^4 \times 8 = 128$.

* PLA (Prog. Logic Array):

$\Rightarrow$ Prog. AND gates + Prog. OR gates.



$F_1(A, B, C) = A\bar{C} + \bar{B}\bar{C}$

$F_2(A, B, C) = \bar{A}B + A\bar{C}$.

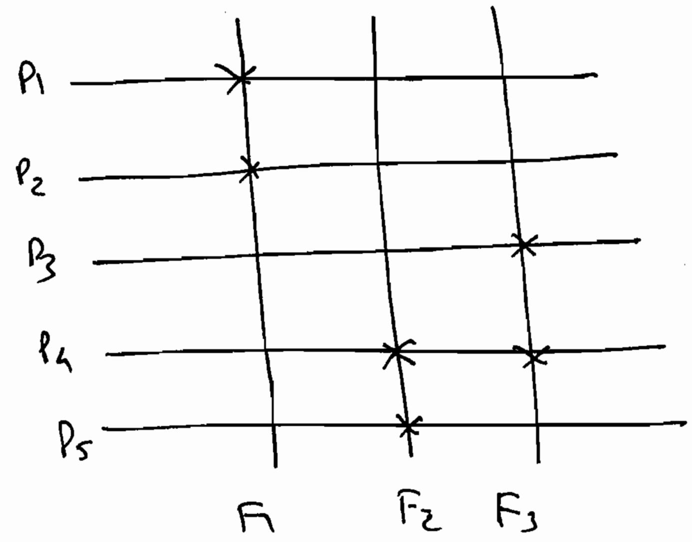
Product terms $\longrightarrow$ $P_1 = A\bar{C}$

$P_2 = \bar{B}\bar{C}$

$P_3 = \bar{A}B$.

PLA size $\Rightarrow$ (3 inputs, 3 Product terms, 2 outputs).

Ex-1 In the following PLA Determine the product terms P_1, P_2, P_3, P_4, P_5



$$F_1(A,B,C) = \bar{A}B\bar{C} + A\bar{B}C + AB.$$
$$= \bar{A}B\bar{C} + (\bar{B}C + B)A$$
$$= \bar{A}B\bar{C} + AC + AB.$$
$$= B(\bar{A}\bar{C} + A) + AC$$
$$F_1(A,B,C) = \cancel{\bar{A}}B\bar{C} + AB + AC = \cancel{A}B\bar{C} + AC$$

$$F_2(A,B,C) = \bar{A}\bar{B}\bar{C} + ABC + \bar{A}C$$
$$= \bar{A}\bar{B}\bar{C} + C(AB + \bar{A})$$
$$= \bar{A}\bar{B}\bar{C} + BC + \bar{A}B.$$
$$= \bar{A}(\bar{B}\bar{C} + B) + BC$$
$$= \bar{A}(\bar{C} + B) + BC$$
$$F_2 = \bar{A}\bar{C} + \bar{A}B + BC = \bar{A}\bar{C} + BC$$

$$F_3 = \bar{A}B\bar{C} + BC + A\bar{B}C.$$
$$= \bar{A}B\bar{C} + C(A + B).$$
$$= AC + BC + \bar{A}B\bar{C}$$
$$= AC + B(C + \bar{A})$$
$$= AC + BC + \bar{A}B.$$
$$= AC + \bar{A}B.$$

$F_1 =$

A \ BC	00	01	11	10
0				1
1		1	1	1

$F_1 = \underline{AC} + B\bar{C}$

$F_2 =$

A \ BC	00	01	11	10
0	1	1	1	
1			1	

$F_2 = \bar{A}\bar{B} + BC.$

$F_3 =$

A \ BC	00	01	11	10
0			1	1
1		1	1	

$F_3 = \underline{AC} + \bar{A}B.$

$\therefore \quad P_2 = AC \quad , \quad P_1 = B\bar{C} \quad , \quad P_3 = \bar{A}B.$

$P_4 = \bar{A}B \quad , \quad P_5 = BC.$ or

$P_5 = \bar{A}B \quad , \quad P_4 = BC.$

size = $(3, 5, 3)$

(i/p, product term, o/p)

* PAL : Prog. AND gate + Fixed OR gate.

GAL : Generic Array logic

= Prog. AND gates + fixed OR gate + Prog output Logic (E.g. Tri state O/P, Normal o/p etc).

Ex-1 How many invalid i/p combinations occur at input of BCD adder?

Ans: No. of Invalid BCD Combination at Input

= Total Input Combinations − No. of valid Combinations.

$= (16 \times 16) - (10 \times 10)$

$= 156.$

* Over flow:

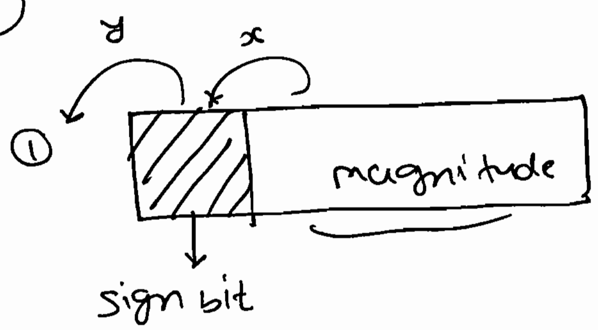
→ overflow is used in signed arithmatic.

→ overflow occurs whenever the result exceeds the range of signed no.s.

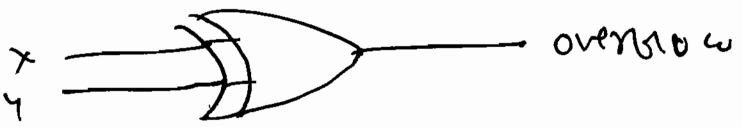
→ The overflow condition can be varified by the following methods:

(1)

→



If $x = 1$ and $y = 0$ (or) $x = 0$ and $y = 1$ } → overflow occured.



② Overflows occurs if 2 +ve no.s are added the result is -ve or 2 -ve no.s are added the result is +ve.

E.g. Consider 2's comp. nos.

(a)
$x=0$, $y=1$

1.0 1	(-3)
+ 1.1 0	(-2)
~~1~~ 0 ~~1~~ 1	

Overflow occured (or) Two -ve nos are added result is +ve

(b)
$y=0$, $x=1$

0 1 0	(2)
+ 0 1 1	(3)
1 0 1	(-3)

Overflow occured (or) Two +ve nos are added Result is -ve

(c)
$y=1$, $x=1$

1 1 0	(-2)
+ 1 1 1	(-1)
~~1~~ 1 0 1	~~(-2)~~ (-3)

No overflow

Ex-1

$$\begin{array}{r} x \\ +\ y \\ \hline z \end{array}$$

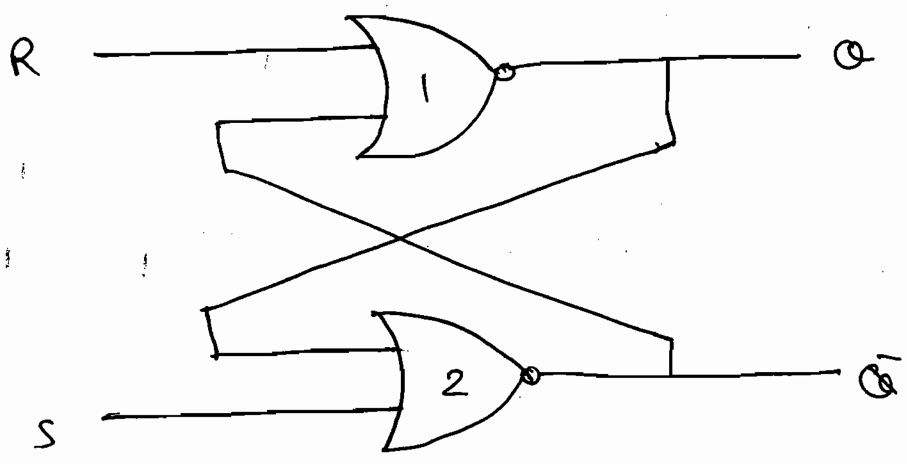
overflow

If $x=1$, $y=1$ but $z=0$
(or)
$x=0$, $y=0$ but $z=1$.

Expression for overflow i.e. is $\Rightarrow$ $\boxed{\bar{x}.\bar{y}\,z + x.y.\bar{z}}$

☆ Sequential Circuits : (1 bit memory elements)

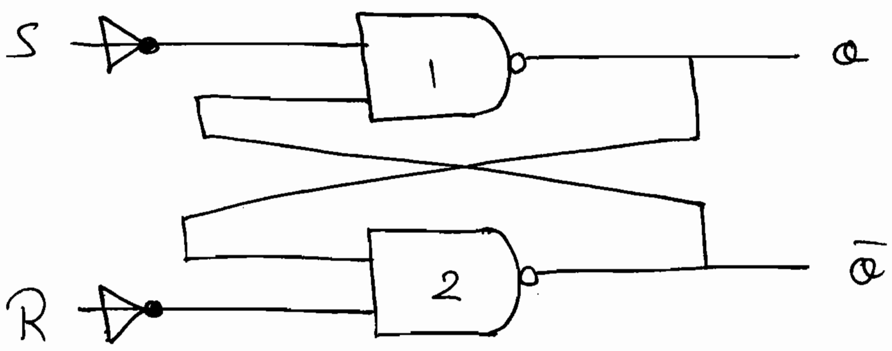
⇒ * S-R Latch:



S	R	Q
0	0	No change
0	1	'0'
1	0	'1'
1	1	Impractical state

(∵ $Q = \bar{Q} = 0$).

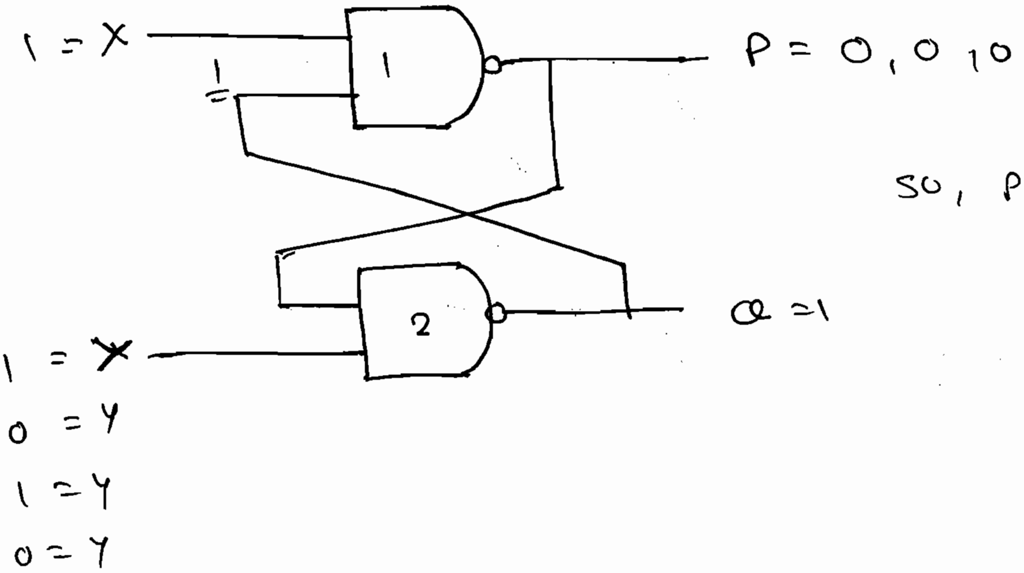
⇒



*

S R	Q
~~01~~	$\boxed{1}$
00	1

Ex-1 In the following X-Y Latch initially x=1 & y=1. Determine the outputs P & Q If the y input is change as 0,1,0,1,0,1,....

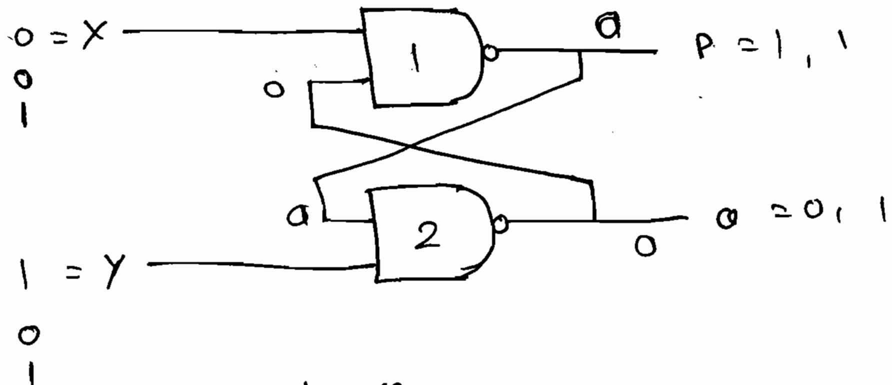


so, p is fixed at p=0, Q=1.

Ex-2 In the following X-Y Latch the seq. of inputs are XY = 01, 00 and 11. determine the values of P & Q.

10, 1ϕ, QQ.

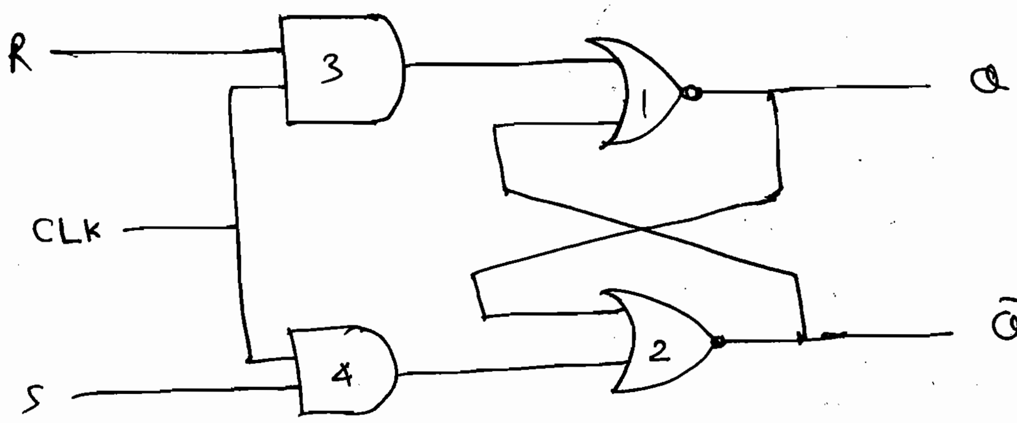
10, 11, 0ϕ.

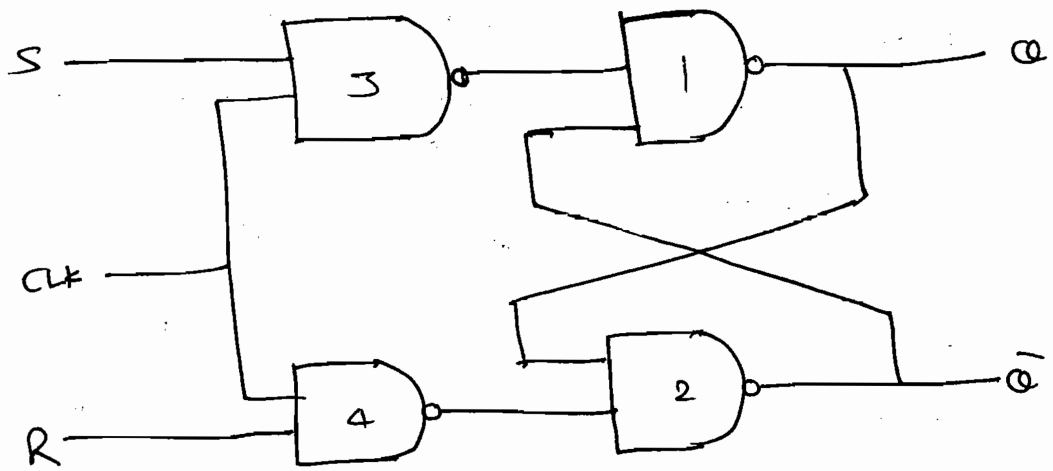


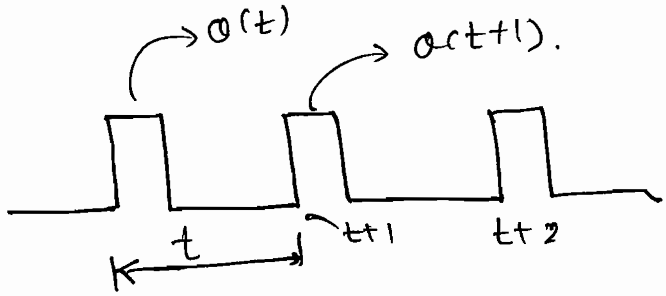
X Y	P Q
0 1	1 0
0 0	1 1
1 1	0 1 ← If gate 1 is faster than gate 2. (OR) ϕ0 ← If gate 2 is faster than gate 1.

* Clocked S-R flip flop:

→ Synchronized latch is called flip-flop.







T = Clock period

$\frac{1}{T} = f =$ Clock freq.

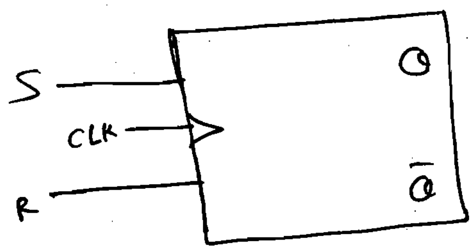
* Truth table:

S	R	Q(t+1)
0	0	Q(t)
0	1	0
1	0	1
1	1	Ambiguous state

do not apply.

* Characteristic Table:

S	R	Q(t)	Q(t+1)
0	0	0	0
0	0	1	ϕ
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	X
1	1	1	X



S \ R Q(t)	00	01	11	10
0		1		
1	1	1	X	X

→ S

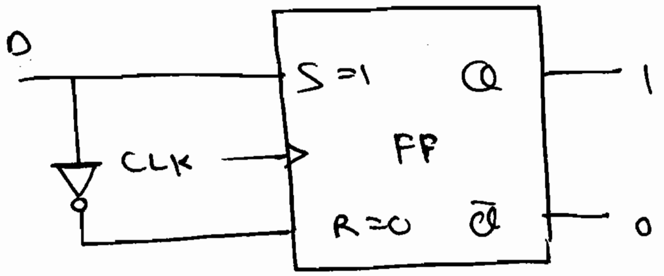
$\bar{R}\,Q(t)$.

$$\boxed{Q(t+1) = S + \bar{R}\,Q(t)}$$

* Clocked D-flip flop:

→ used in shift Register

→ D → Data, Delay.



* Truth table:

D	Q(t+1)
0	0
1	1

S=0
R=1

S=1
R=0

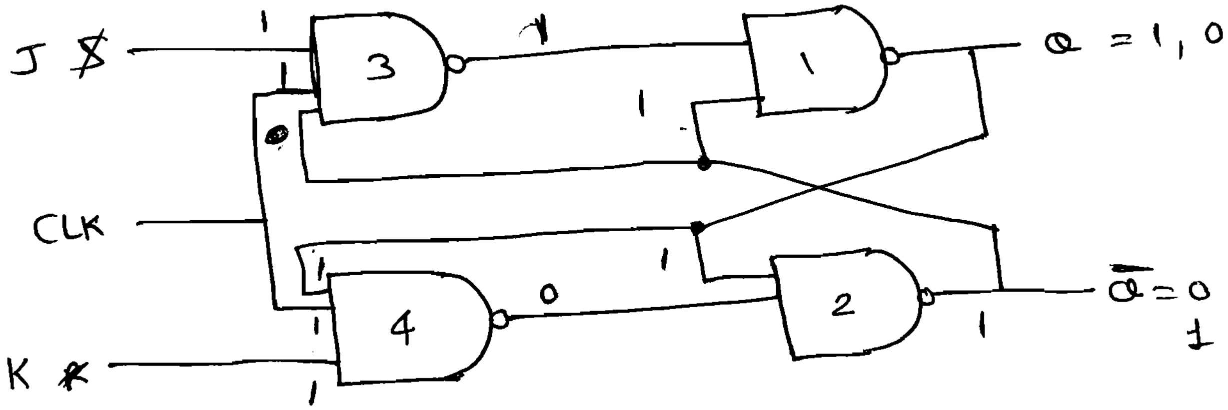
* Characteristic table:

D	Q(t)	Q(t+1)
0	0	0
0	1	0
1	0	1
1	1	1

* Characteristic equation:

$$\boxed{Q(t+1) = D}$$

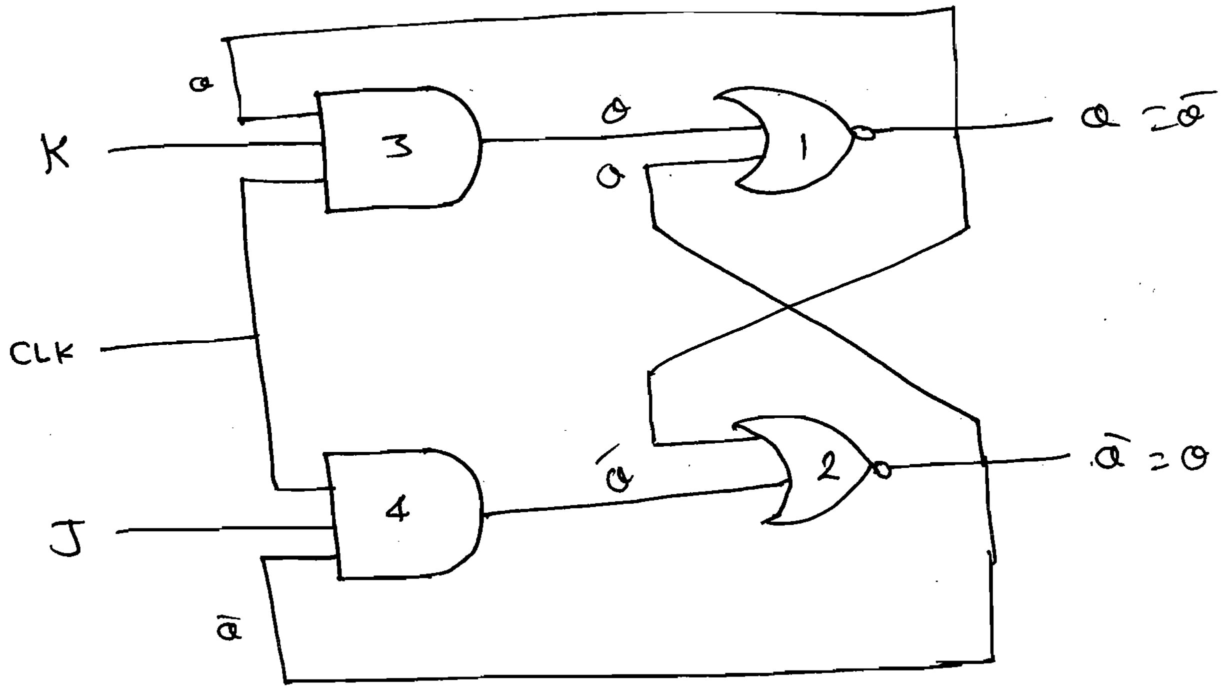
* Clocked J K - Flip Flop.



$S = J \cdot \bar{Q}$

$R = K \cdot Q$

(or)



* Truth table:

S J	R K	Q (t+1)
0	0	Q (t)
0	1	0
1	0	1
1	1	$\overline{Q(t)}$

* Characteristic table:

J	K	Q(t)	Q(t+1)	
0	0	0	0	} Q(t)
0	0	1	1	
0	1	0	0	} 0
0	1	1	0	
1	0	0	1	} 1
1	0	1	1	
1	1	0	1	} $\overline{Q(t)}$
1	1	1	0	

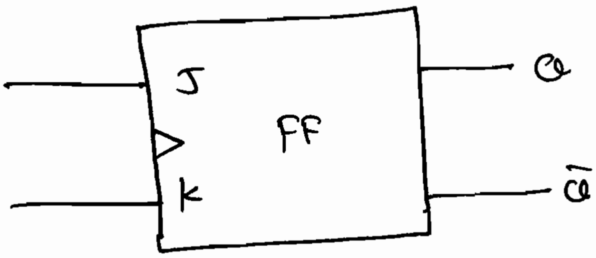
toggle

* Characteristic ~~xxxxx~~: equation:

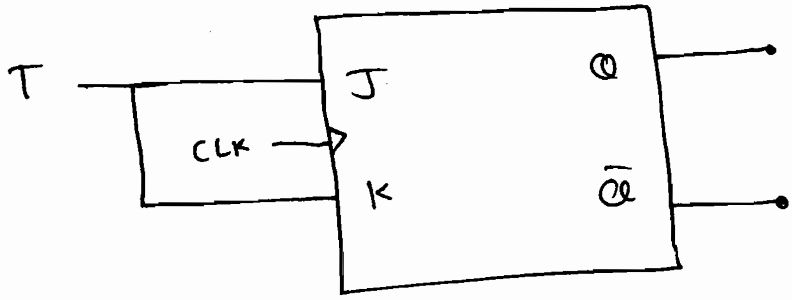
J \ KQ(t)				
		1		
	1	1		1

$$\therefore \boxed{Q(t+1) = J \cdot \overline{Q(t)} + \overline{K} \cdot Q(t)}$$

*



* Clocked T - Flip-Flop:



* Truth Table:

T	Q(t+1)
0	Q(t)
1	$\overline{Q(t)}$

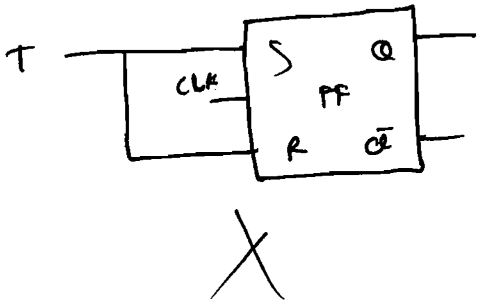
* Characteristic Table:

T	Q(t)	Q(t+1)	
0	0	0	Q(t)
0	1	1	
1	0	1	$\overline{Q(t)}$
1	1	0	

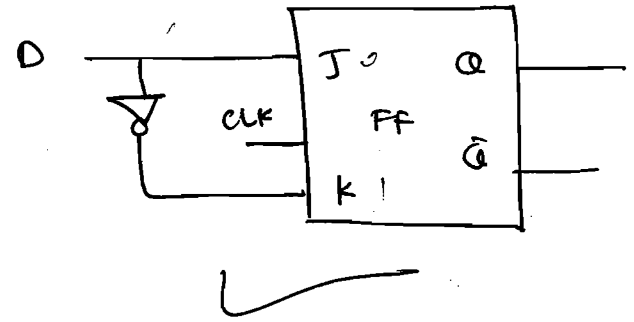
* Characteristic eqn:

$$\boxed{Q(t+1) = T \oplus Q(t)}$$

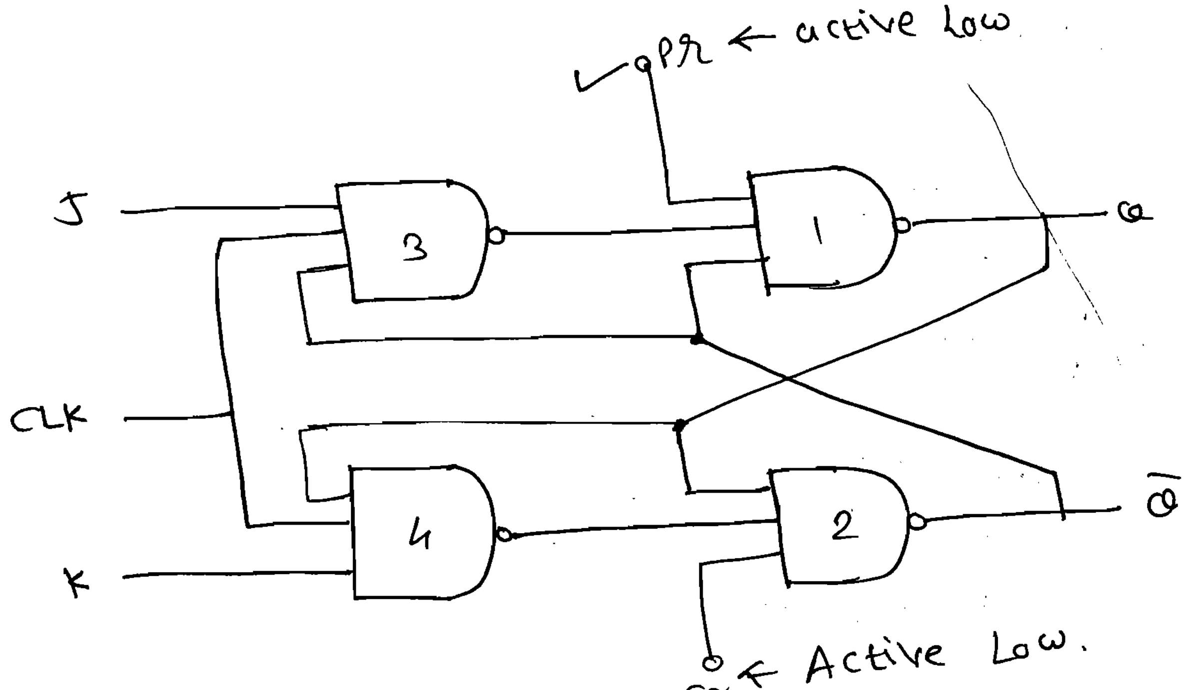
(a)

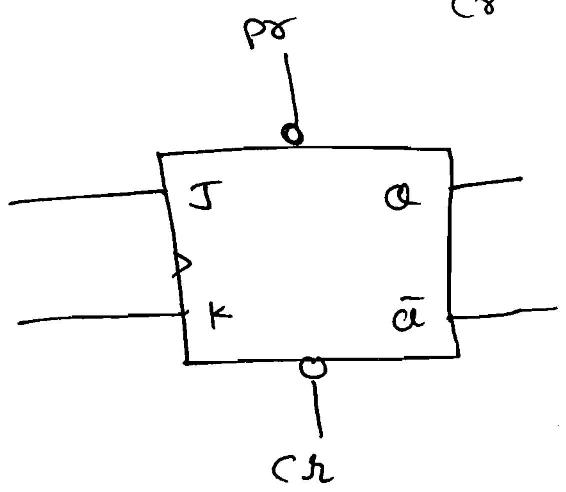


(b)



* Asynchronous (or) Direct Inputs → preset (Pr), clear (Cr).





CLK	Pr	Cr	Q
0	0	1	1
0	1	0	0
⎍⎍⎍	1	1	depends on FF I/P.

* Setup, Hold times of FF.

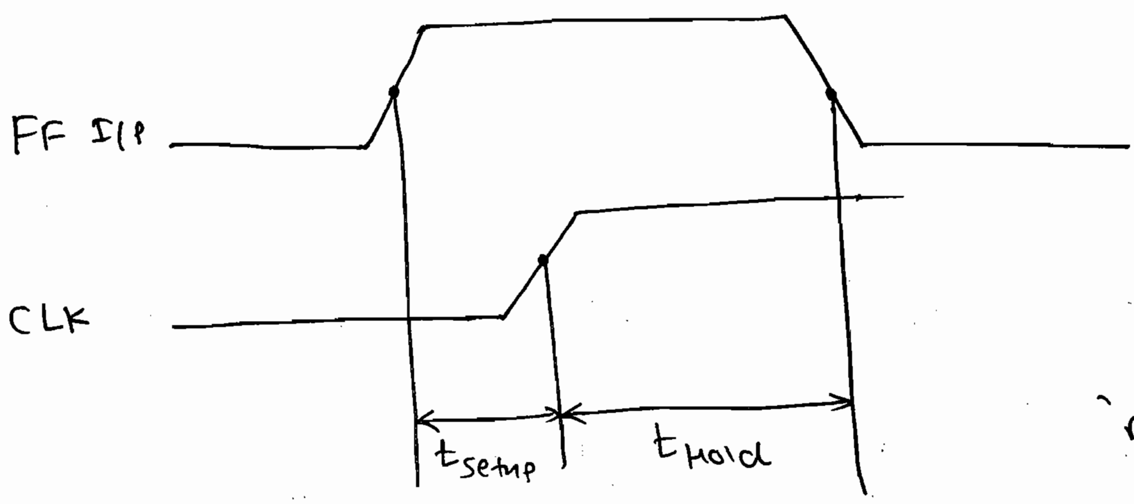
(1) Setup time:

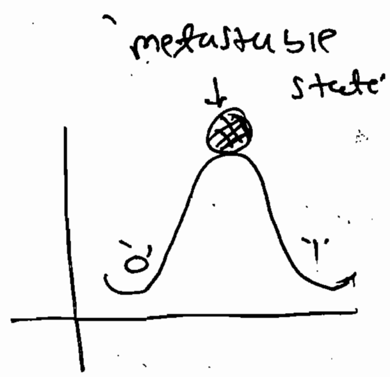
→ It is the minimum time by which the Input should come ahead of the clocked input.

(2) Hold time:

→ It is the minimum time for which the input should be maintained constant after applying the clock pulse.

→ If setup time and hold times are not satisfies then the FF enters into metastable state i.e. neither zero nor '1' output.

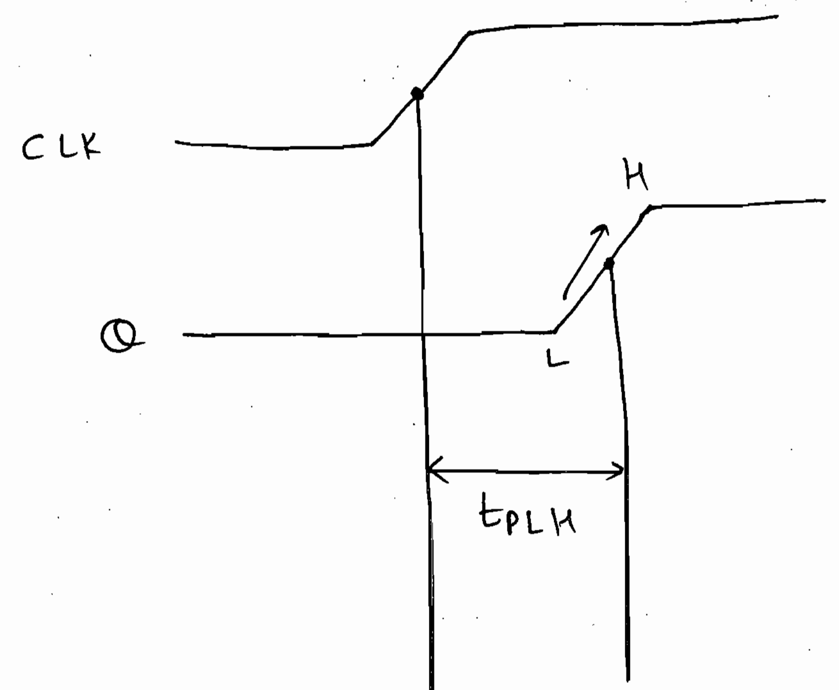




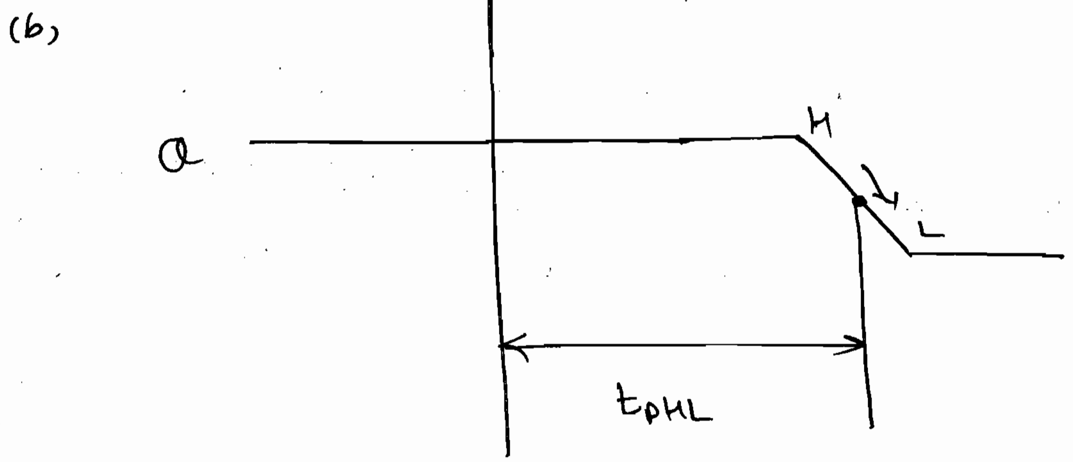
* Propogation Delay for FF.

→ It is the time taken by the flip flop to change it's state. (i.e. from L → H (or) H → L.)

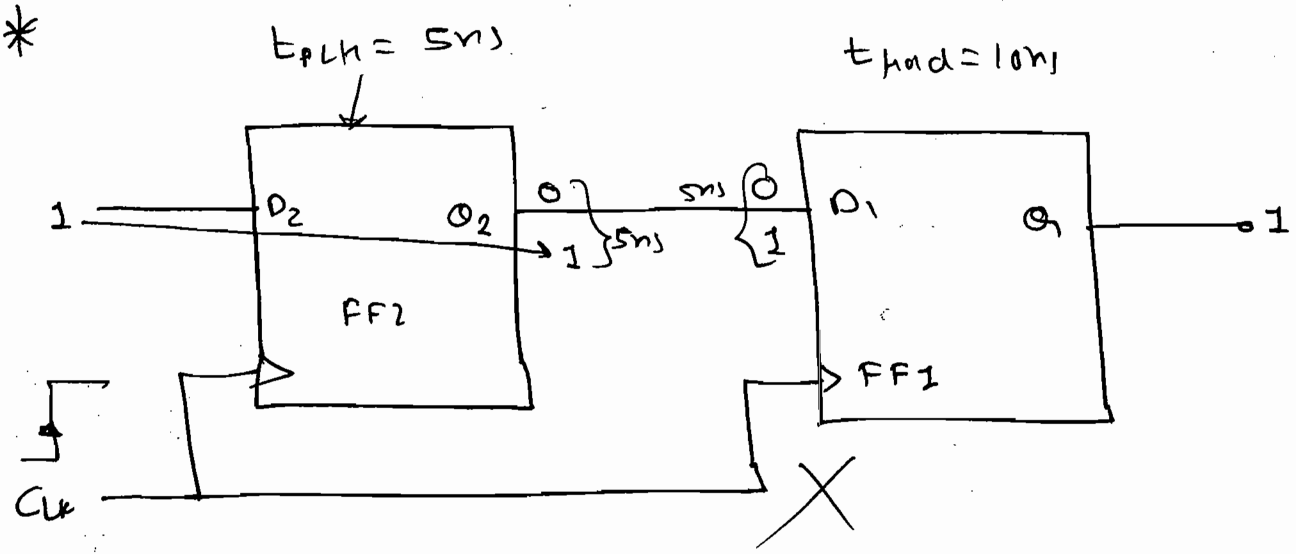
(a)



$$\therefore \boxed{t_p = \frac{t_{pLH} + t_{pHL}}{2}}$$



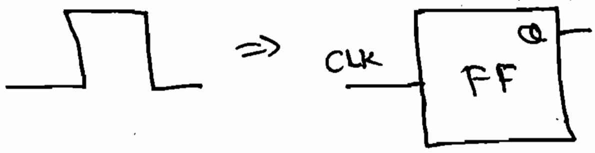
*



→ For proper FF operation $\boxed{t_{Hold} < t_{pLH}, t_{pHL}.}$

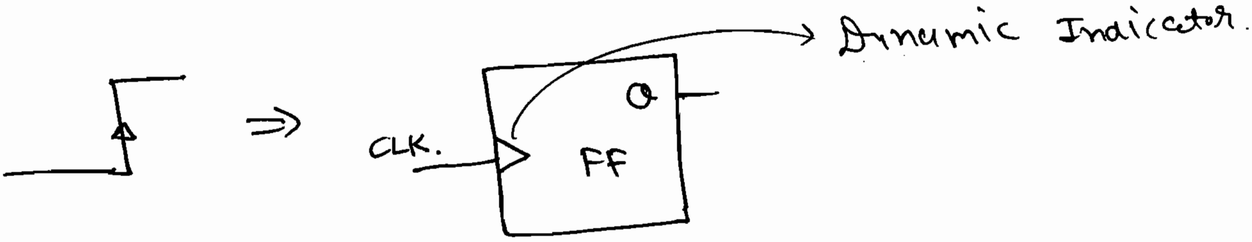
* Types of Triggering:

1) Level Triggered FF

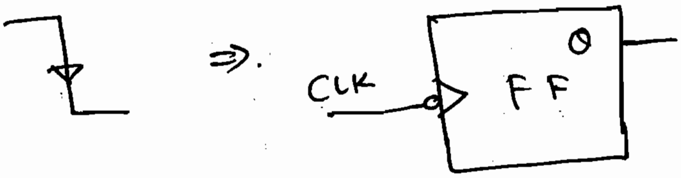


2) Edge Triggered FF:

(a) Positive Edge Triggered FF
(Leading Edge Triggered FF).

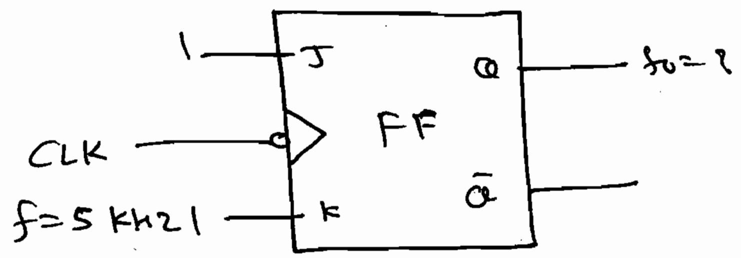


(b) Negative Edge Triggered FF
(Trailing Edge Triggered FF).



* Determine the o/p freq. of following FF if the CLK freq is 5 KHz.

(a)



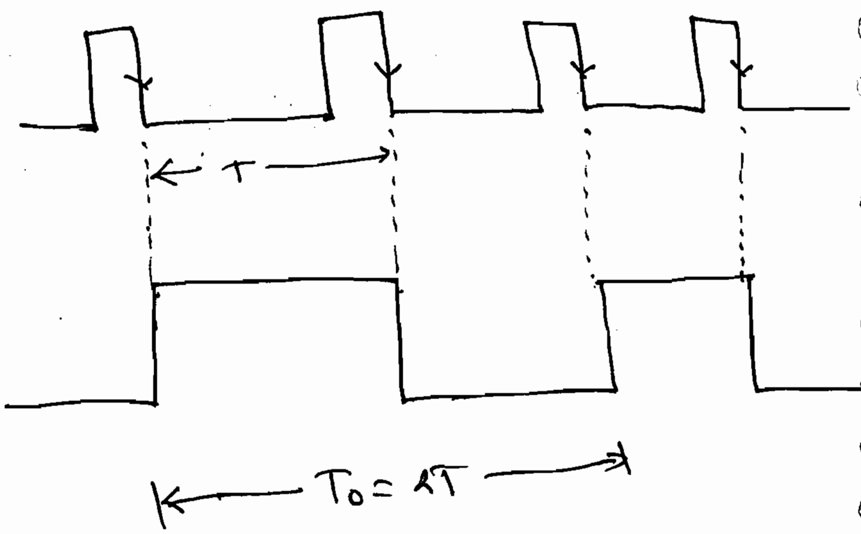
NOTE:

→ If a FF is in toggle mode the o/p freq is half of the CLK freq. ~~Unit~~ with 50% duty cycle. (i.e) A toggle mode FF will act as freq. divider by 2.

→ $J = K = 1 \Rightarrow$ FF is in Toggle.

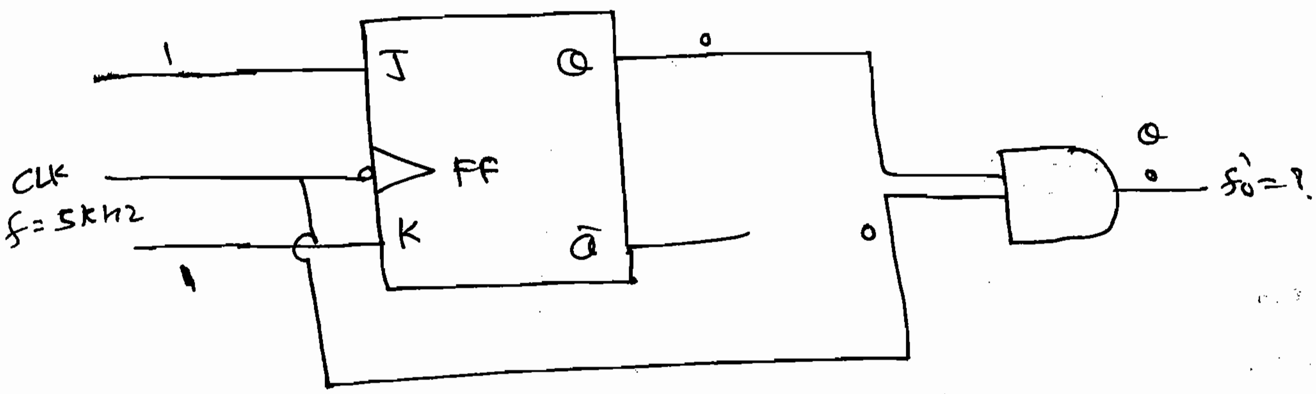
i.e. $\boxed{Q(t+1) = \overline{Q(t)}}$.

CLK	Q
0	0
1	1
2	0
3	1
4	0



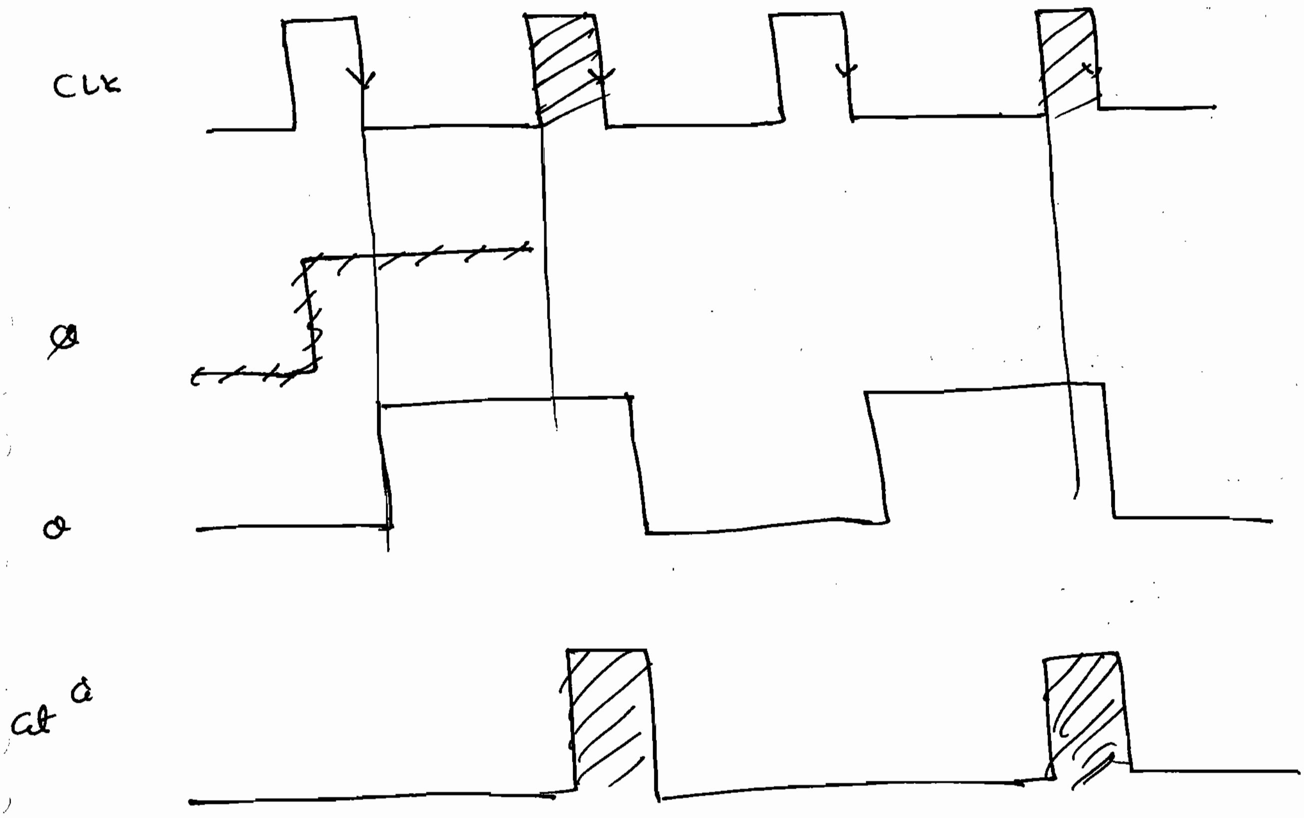
$$\frac{1}{T_0} = \frac{1}{2T} \Rightarrow \boxed{f_0 = \frac{f}{2}} \quad \therefore f_0 = 2.5 \text{ KHz}.$$

(b)



CLK	O/P at Q
0	1
1	0
2	1
3	0
4	1

→

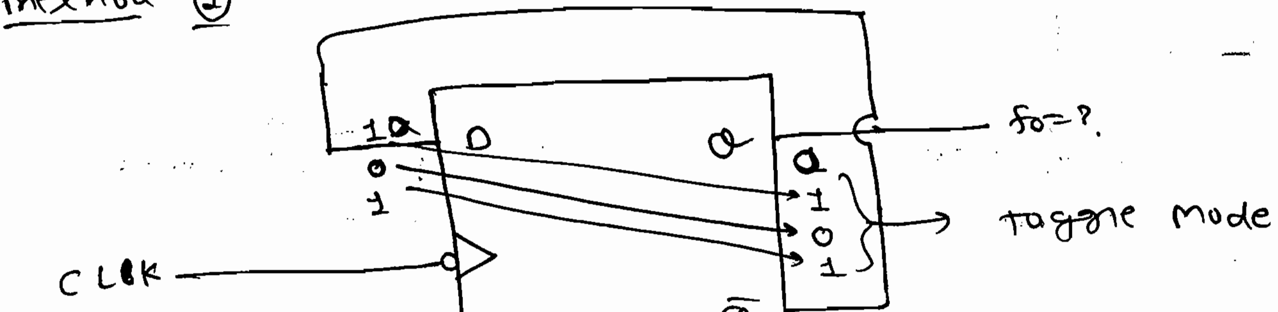


$\longleftarrow T_o' = 2T \longrightarrow$

$\therefore$ $\boxed{f_o = 2.5kHz}$

(c)

* method - (1)



$f_0 = \frac{f}{2}$

$\boxed{f_0 = 2.5\ kHz}$

→

* method - (2)

→ D - FF Char. eqn ⇒ $Q(t+1) = D$. —(1)

But $D = \bar{Q}(t)$. —(2)

Put (2) in (1)

$\therefore \boxed{Q(t+1) = \bar{Q}(t).}$

FF is in "toggle mode"

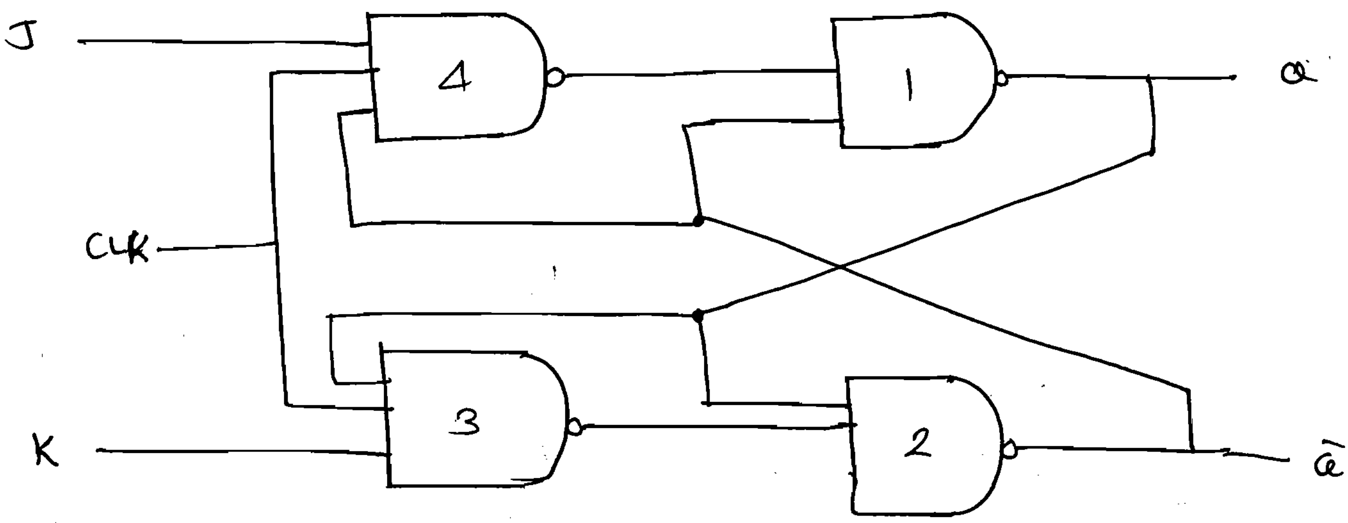
hence $f_0 = \frac{f}{2}$

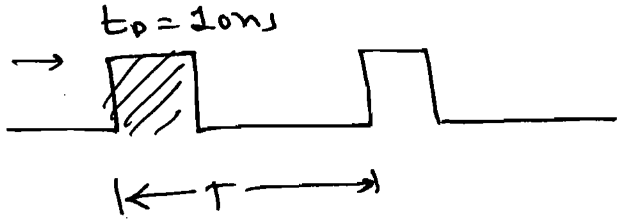
$\therefore \boxed{f_0 = 2.5\ kHz}$

* Race Around Condition: (RAC).

→ RACE AROUND CONDITION occurs in Level trigger flip flop and doesn't occur in edge triggered FFs.

→ $\Delta t =$ FF prop. delay $= 2ns$



→ 

⇒ "RAC" is when $J = K = 1$ and $t_p \gg \Delta t$.

→ Output Toggels manytimes instead of once.

* How to avoid RAC (in level Triggered FF).

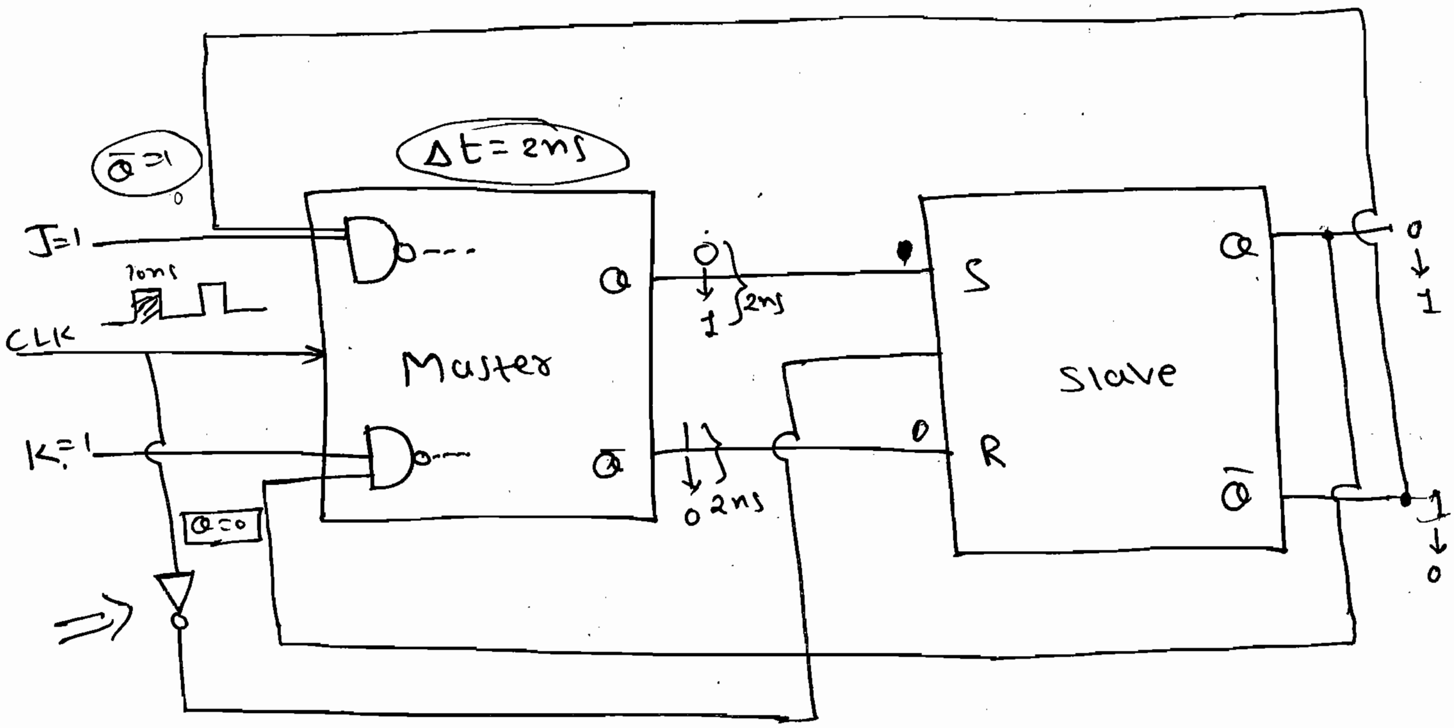
① Choose FF prop delay 'Δt' such that.

$$\boxed{t_p < \Delta t < T}$$

② Master- Slave JK FF.

* Master slave JK FF

→ In master slave JK FF the feedback values Q and $\bar{Q}$ do not change during the clock pulse. eventhough the output changes. Hence, 'RAC' doesn't occur.

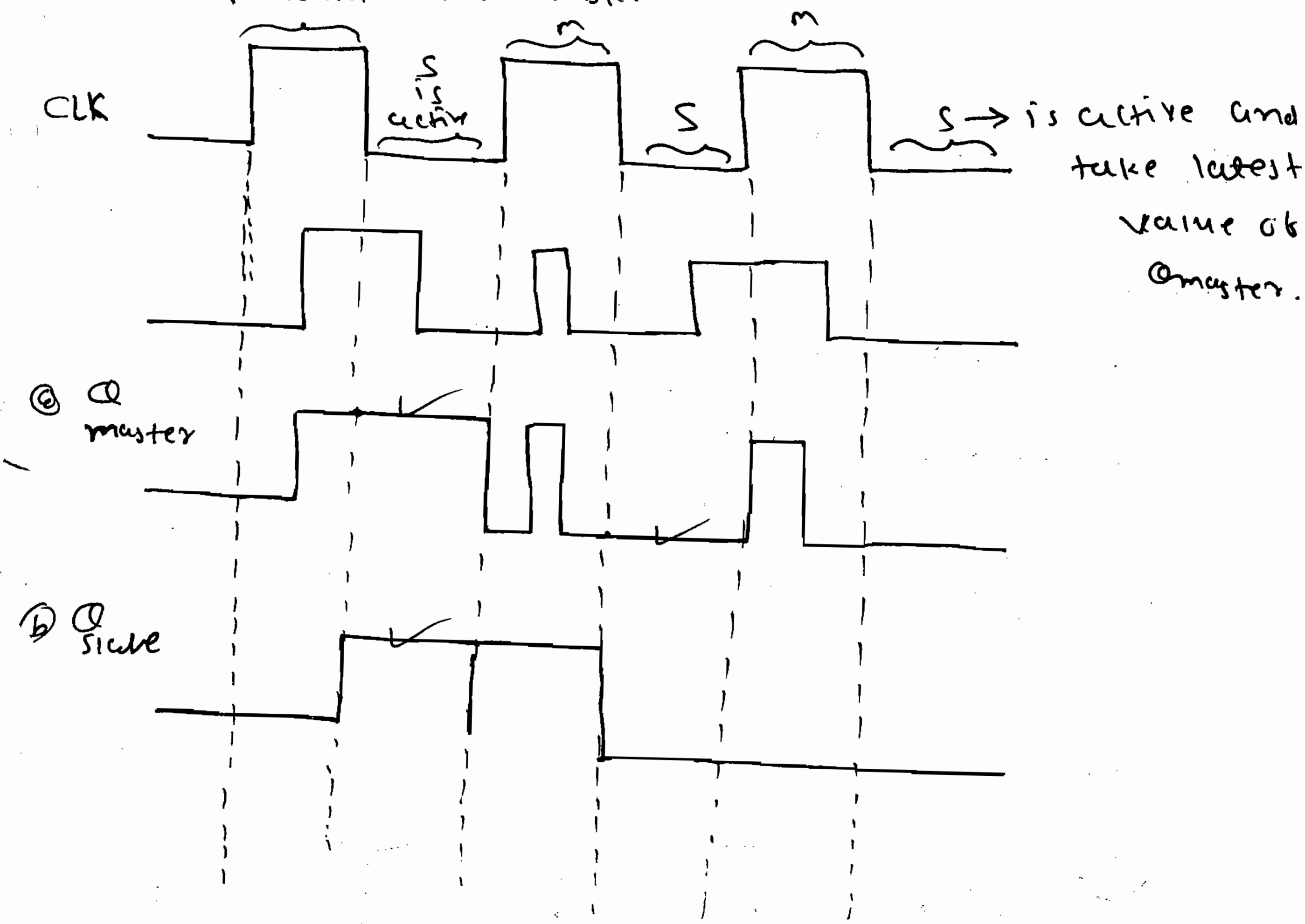


→ In master slave J-K flip flop the feedback values Q and $\bar{Q}$ do not change during the clock pulse because they are taken from inactive slave FF. Hence 'RAC' doesn't occured.

→ Master slave JK FF output is similar to the negative edge triggered JK flip flop output.

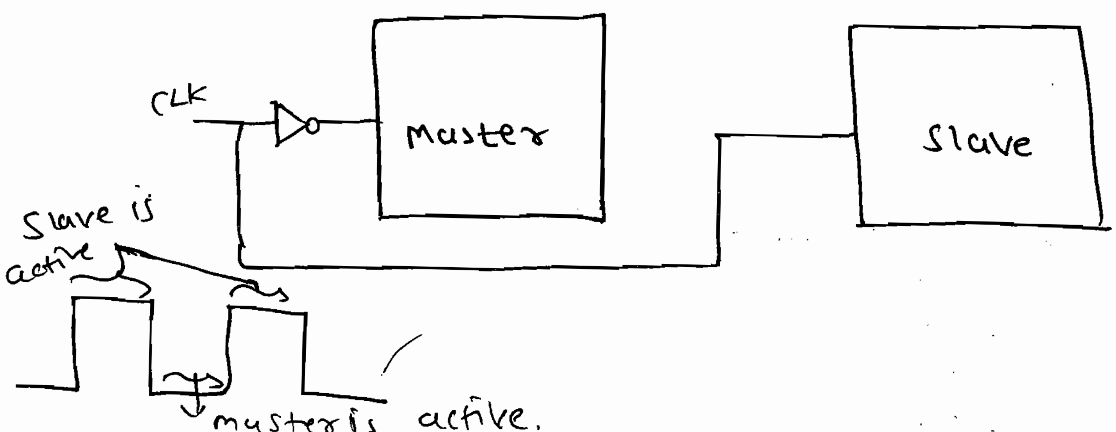
Ex-1 In a master slave D flip flop Draw the o/p of master and slave if the clock and D inputs are as given below.

→ m is active and Qmaster = D



→ Slave o/p is similar to -ve Edge Triggered FF o/p.

NOTE: Master slave JK FF as JKFF as Positive Edge triggered FF.



→ In slave we can use J-K flip flop instead of S-R flip flop. But is more costly than S-R.

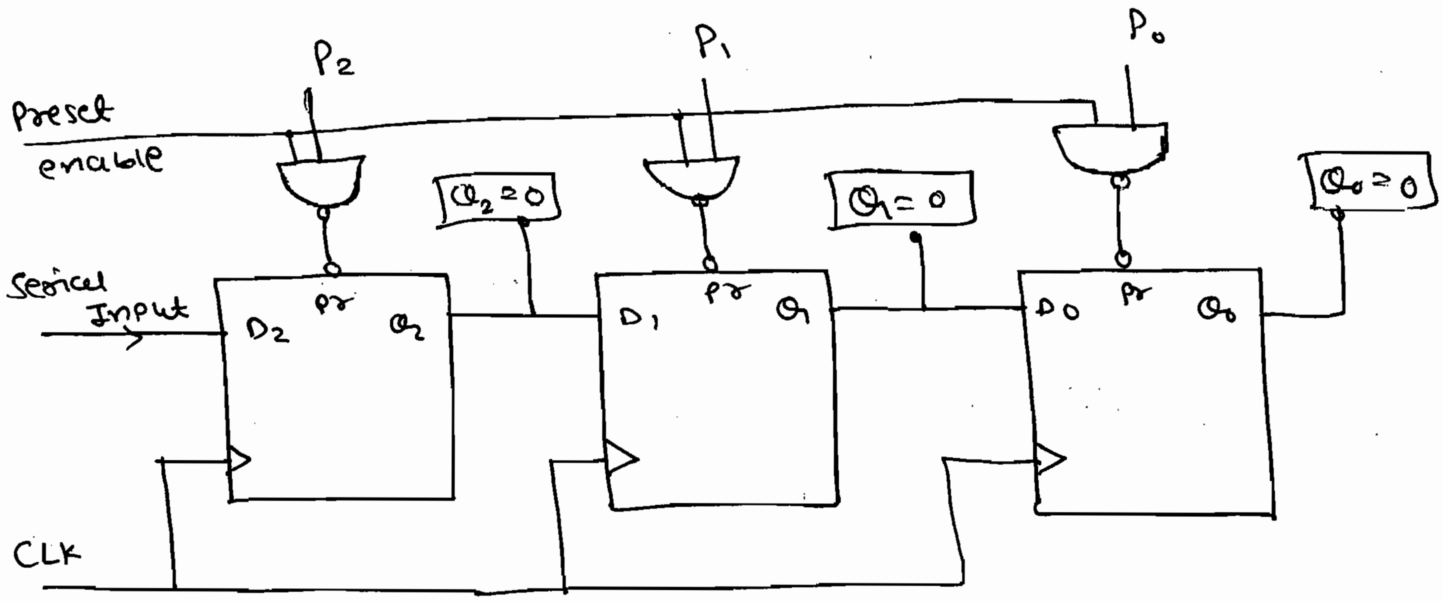
① Shift Register → "Sequential memory".

② Counters

ⓐ to count the no. of pulses.

ⓑ Frequency Divider.

* 3-Bit Shift Register:



⇒ Serial I/P bit → Q_2

$Q_2 \to Q_1$

$Q_1 \to Q_0$

① Serial in Parallel out :

CLK	Serial Input	Q_2	Q_1	Q_0
0	–	0	0	0
1	LSB 1	1	0	0
2	0	0	1	0
3	1	1	0	1

serial → parallel

MSB LSB
101

* 'N' Bit shift Register.
  'N' clock pulses. $\boxed{\text{time} = \boxed{N \cdot T}}$ secs.

② Serial in serial out:

101 → SISO → 101.

CLK	Serial Input	Q_2	Q_1	Q_0
0	–	0	0	0
1	LSB 1	1	0	0
2	0	0	1	0
3	1	1	0	1
4	–	–	1	0
5	–	–	–	1

Serial Input

serial o/p.

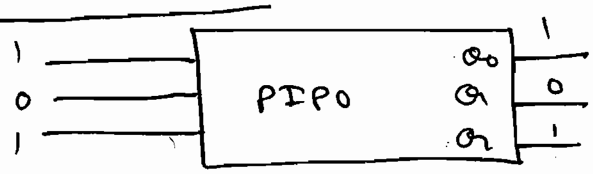
$N + (N-1) = (2N-1)$ CLK periods.

$\boxed{\text{Time} = (2N-1)\,T.}$

N = No. of FFs.

③ Parallel in parallel out:

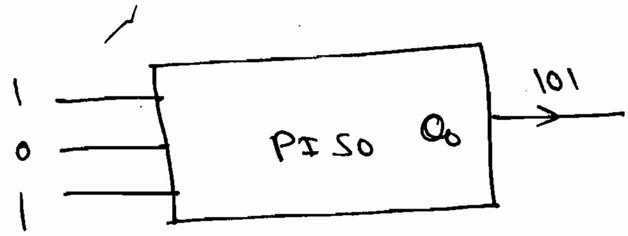
$P_2 P_1 P_0$ = Parallel input



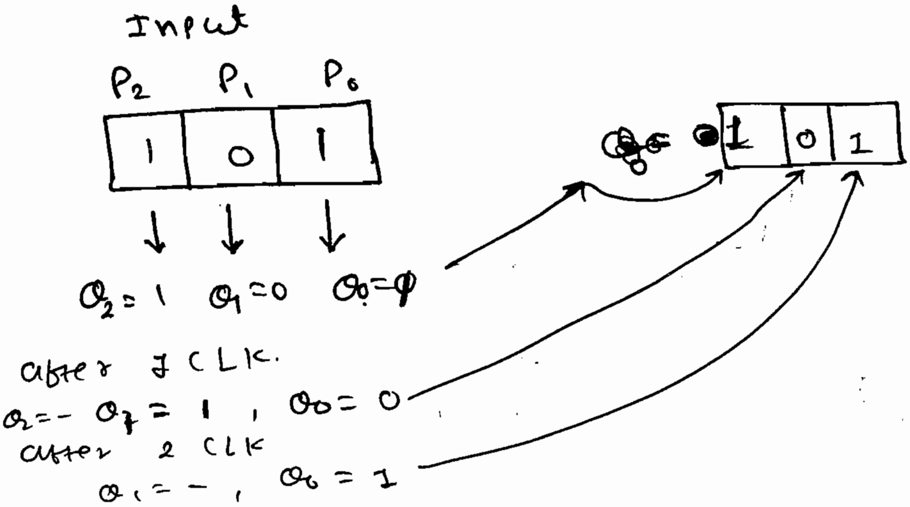
→ No clock pulse is required.

→ Preset is asynchronous input therefore no clock pulse is required.

④ Parallel in serial out:

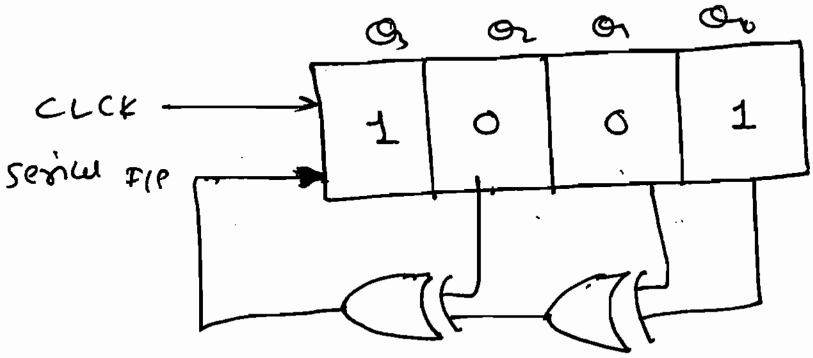


⇒ $\boxed{\text{Time} = (N-1)T}$



NOTE: In parallel in serial out operation preset enable is disabled after obtaining the input data at Q_2, Q_1, Q_0.

Ex-1 In following shift register determine the no. of CLK pulses required to bring the shift register to the initial value of 1001.



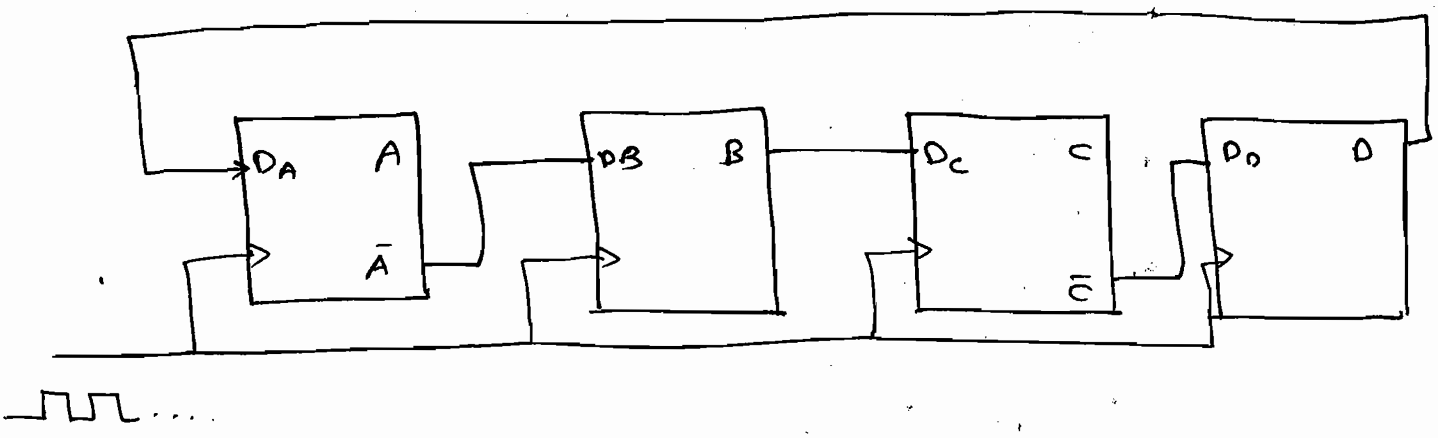
Sr. Input $= Q_2 \oplus Q_1 \oplus Q_0$.

CLK	serial Input $Q_2 \oplus Q_1 \oplus Q_0$	Q_3	Q_2	Q_1	Q_0	
0	–	1	0	0	1	→ 1
1	1	1	1	0	0	→ 1
2	1	1	1	1	0	→ 0
3	0	0	1	1	1	→ 1
4	1	1	0	1	1	→ 0
5	0	0	1	0	1	→ 0
6	0	0	0	1	0	→ 1
7	1	1	0	0	1	

So, 7 clock pulse is required.

Ex-2

Initial value: 0000



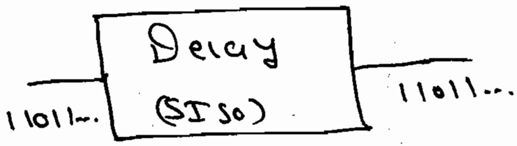
Clock pulse	A $= D$	B $= \bar{A}$	C $= B$	D $= \bar{C}$
—	0	0	0	0
1	0	1	0	1
2	1	1	1	1
3	1	0	1	0
4	0	0	0	0

So, After 4 clock pulse.

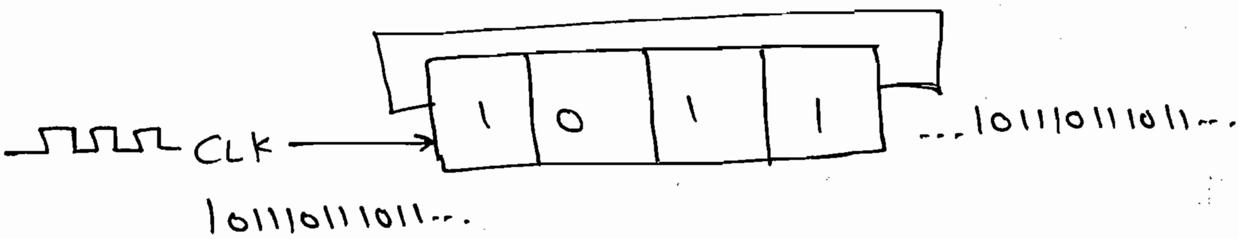
* Shift Register Application:

① Serial to parallel | parallel to serial data converters.

② Time delay



③ Sequence Generator:



④ To generate PN (pseudo Number) sequence.

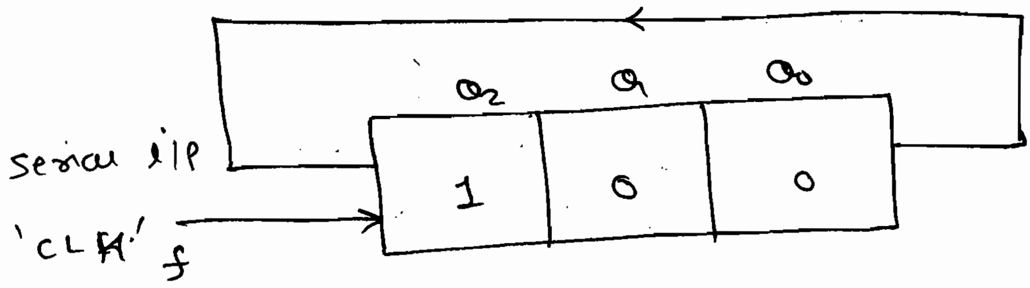
⑤ Counter
ⓐ Ring Counter
ⓑ Johnson Counter.

NOTE:
→ Shift Register is converted to ring counter by making two changes.

① Q_0 is connected to the serial input.

② One of the FF is reset.

* 3 - Bit Ring Counter.



3:1 Counter → Can count 3 CLK pulses.

CLK	serial i/p	Q_2	Q_1	Q_0
0	—	1	0	0
1	0	0	1	0
2	0	0	0	1
3	1	1	0	0

→ Counting:

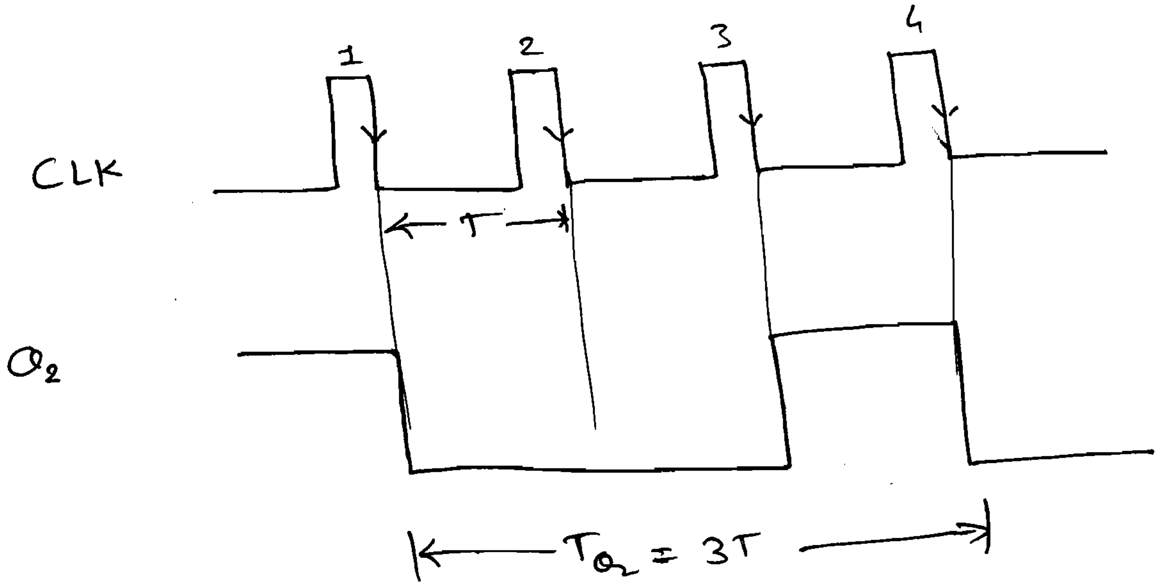
$\boxed{100}$ ← initial value

(a) $\boxed{001} \xrightarrow{\text{Decode}}$ 2 clock pulse.

(b) $\boxed{100} \xrightarrow{\text{Decode}}$ 3 clock pulse.

(c) $\boxed{010} \xrightarrow{\text{Decode}}$ 1 clock pulse.

⇒ Freq division:



$\therefore \quad T_{Q_2} = 3T$

$\therefore \quad \boxed{f_{Q_2} = f/3}$

similarly, $\boxed{f_{Q_1} = f/3}$

$\boxed{f_{Q_0} = f/3}$

⇒ N-bit Ring counter:

→ N:1 Counter

→ Can count N CLOCK PULSE.

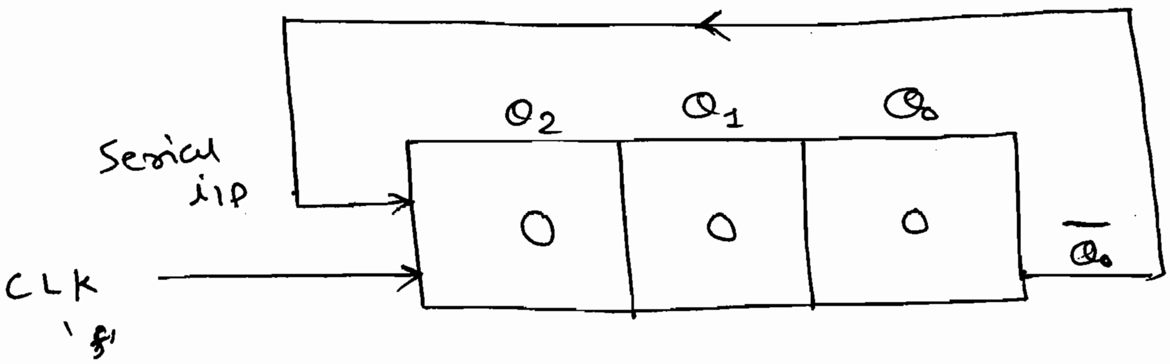
→ Freq. of each FF O/P = f/N.

★ 3-Bit Johnson Counter:

(Twisted Counter (or) Switch-tail counter).

→ Shift Register can be converted into Johnson Counter by making one change.

i.e. $\overline{Q_0}$ is connected to the serial i/p.

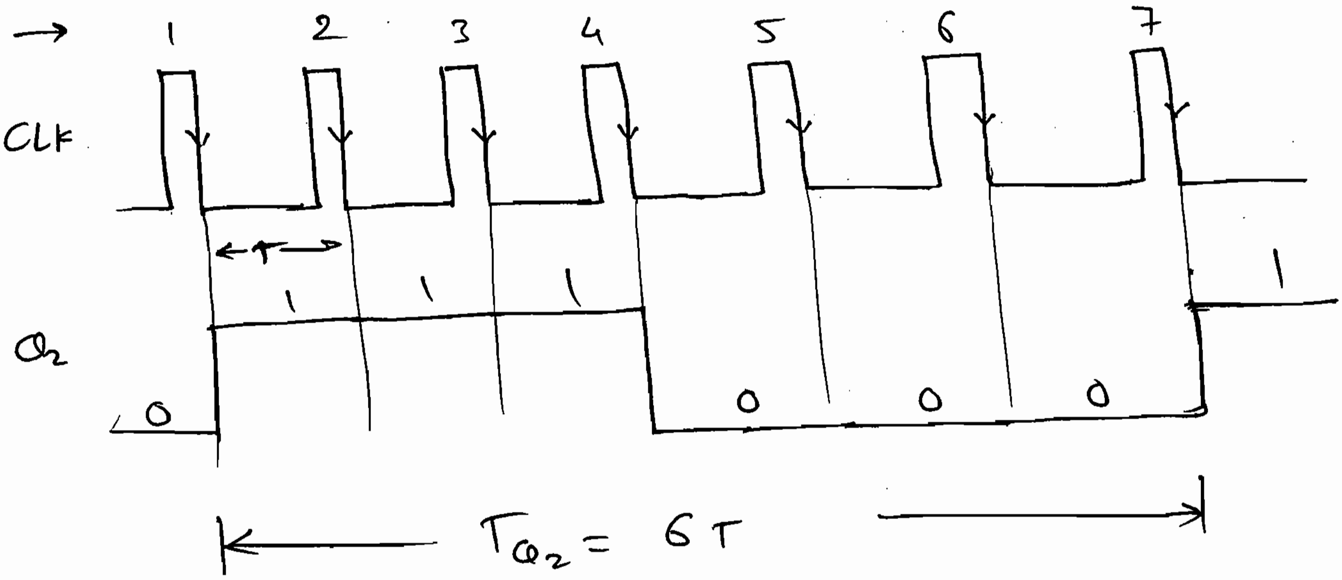


6:1 Counter:

CLK	serial i/p $\overline{Q_0}$	Q_2 Q_1 Q_0	Decimal value
0	—	$\boxed{0\ \ 0\ \ 0}$	0
1	1 →	1 0 0	4
2	1 →	1 1 0	6
3	1 →	1 1 1	7
4	0 →	0 1 1	3
5	0 →	0 0 1	1
6	0 →	$\boxed{0\ \ 0\ \ 0}$	0

⇒ Freq. division:

→



∴ $T_{Q_2} = 6T$

∴ $\boxed{f_{Q_2} = f/6.}$ $\boxed{f_{Q_0} = f_{Q_1} = f/6}$

⇒ Counting:

$\boxed{000}$ ← initial value

a) If $\boxed{001}$ $\xrightarrow{\text{decode}}$ 5 CP.

b) If $\boxed{111}$ $\xrightarrow{\text{decode}}$ 3 CP.

c) If $\boxed{011}$ $\xrightarrow{\text{decode}}$ 4 CP.

⇒ N-bit Johnson Counter:

→ 2N:1 Counter

→ Can count 2N clock pulses.

→ Freq. of each FF o/p = $\frac{f}{2N}$.

Ex 1 Determine the O/P freq. of a 3-bit Johnson Counter. If the clock freq. is 18 kHz. Initial value of the counter is 010.

Ans:

CLK	serial input	Q_2 Q_1 Q_0
0	—	0 1 0
1	1	1 0 1
2	0	0 1 0

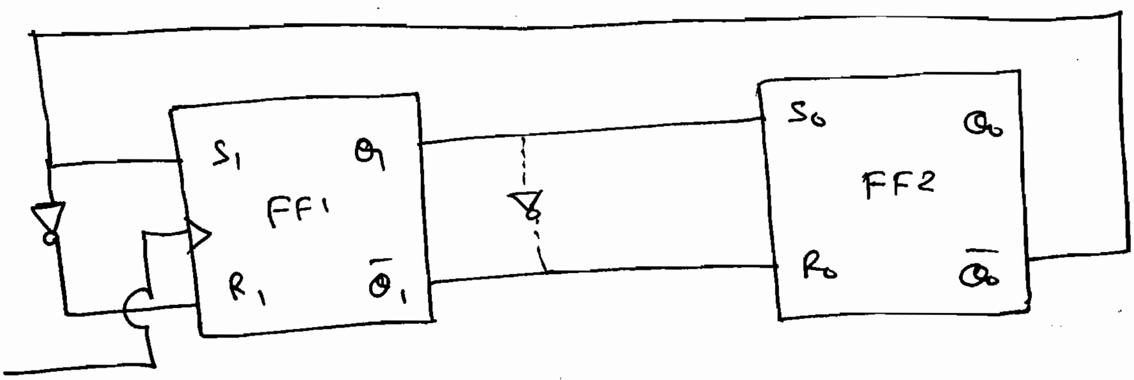
$\therefore$ 2:1 Counter.

$\therefore f_o = f/2$

$\therefore f_o = \frac{18}{2}$.

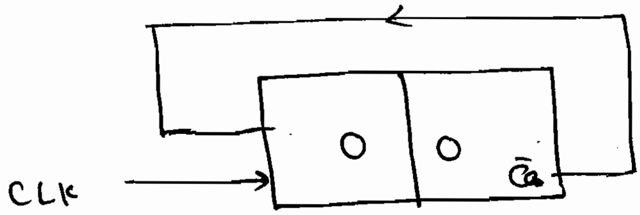
$\therefore \boxed{f_o = 9\ kHz}$

Ex 2 Determine the value of the following ckt after the 729 CP?



→ Given FFs are in toggle mode and are S R FF.

So,



→ The given counters are a 2 bit Johnson counter hence the value of counter after 729 CP is same as the value after 1 CP i.e. $Q_1 = 1$, $Q_0 = 0$

CLK	$\bar{Q}_0$	Q_1 Q_0
0	–	0 0
1	1	1 0

$$
\begin{array}{r}
182 \\
4 \,\overline{)\,729} \\
4 \\
\hline
32 \\
32 \\
\hline
09 \\
8 \\
\hline
1
\end{array}
$$

☆ Counter:

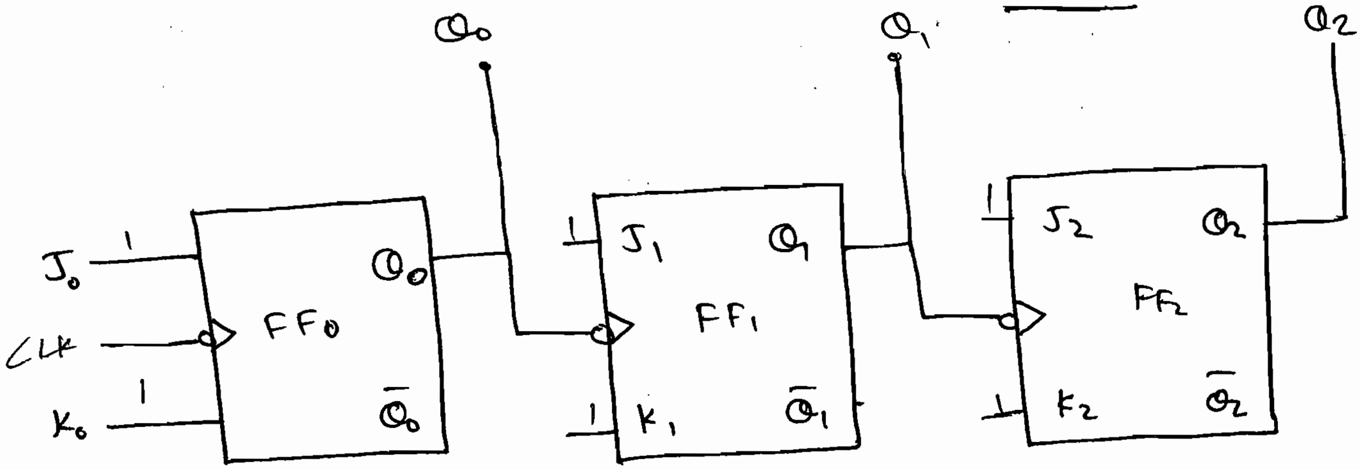
① Asynchronous (Ripple) Counter:

→ Any FF, but in toggle mode:

② Synchronous (parallel) Counters:

→ Any FF

① 3-Bit Asynchronous Counters: (UP Counter) (Ripple counter)



→ As J=K=1 => All FFs are in toggle mode.

$$\overline{Q(t+1)} = Q(t).$$

→ Q_0 is toggle for every clock pulse.

→ Q_1 is toggle when Q_0 changes from '1 to 0'

→ Q_2 is toggle when Q_1 chages from '1 to 0'.

CLK	Q_0 (LSB)	Q_1	Q_2 (MSB)	
0	0	0	0	0
1	1	0	0	1
2	0	1	0	2
3	1	1	0	3
4	0	0	1	4
5	1	0	1	5
6	0	1	1	6
7	~~0~~ 1	1	1	7
8	0	0	0	8

} → Up counter

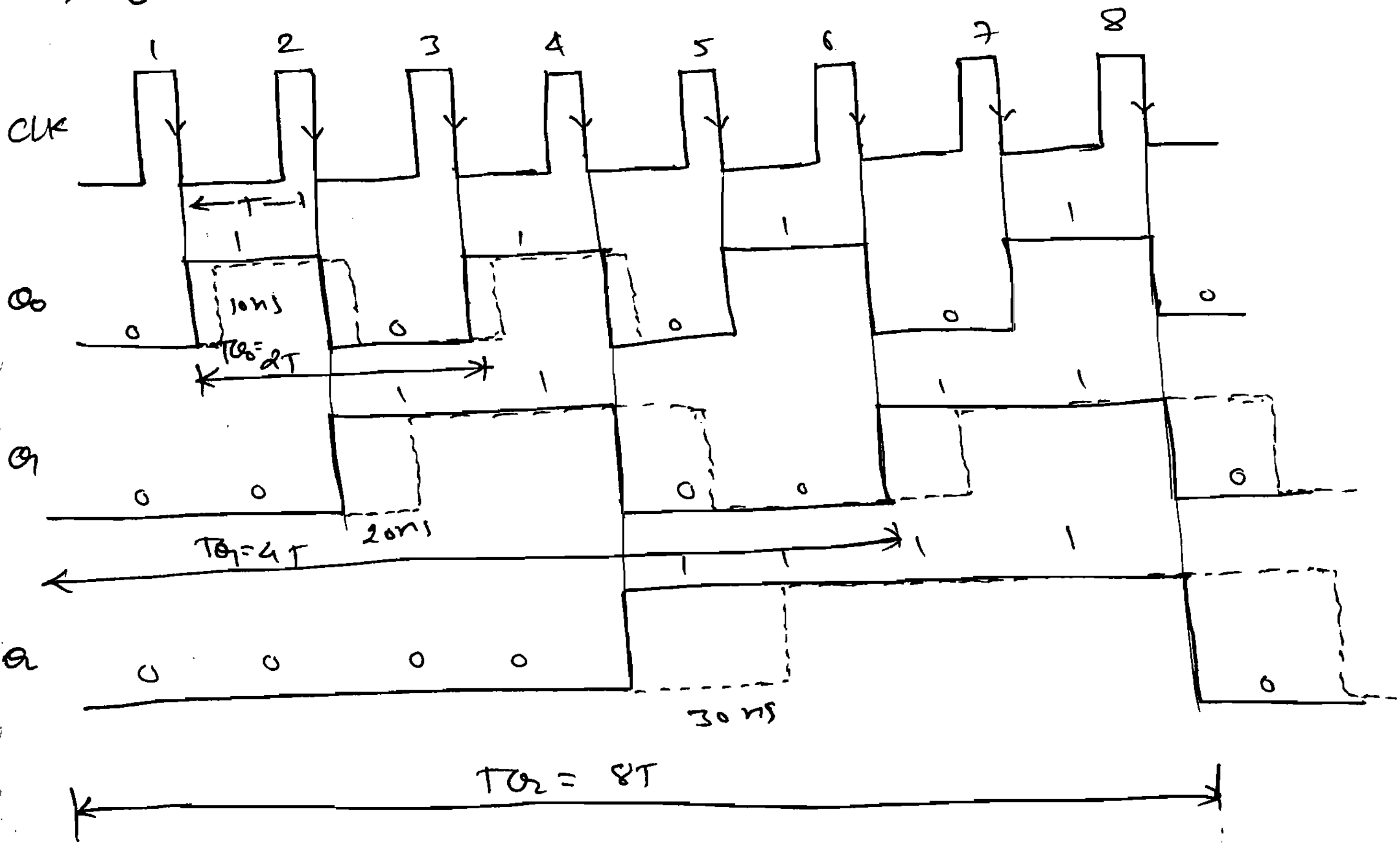
$t_{pd, FF} = 10ns$

Q_0 is delay by 10ns

Q_1 is delay by 20ns

Q_2 is delay by 30ns.

→ freq. is divided by after each flip flop.



* Max. Conv. time = $3 \times 10ns$ = 30ns.

Clock period $T \geq 30ns$.

$\therefore f = \frac{1}{T} \leq \frac{1}{30 \times 10^{-9}}$

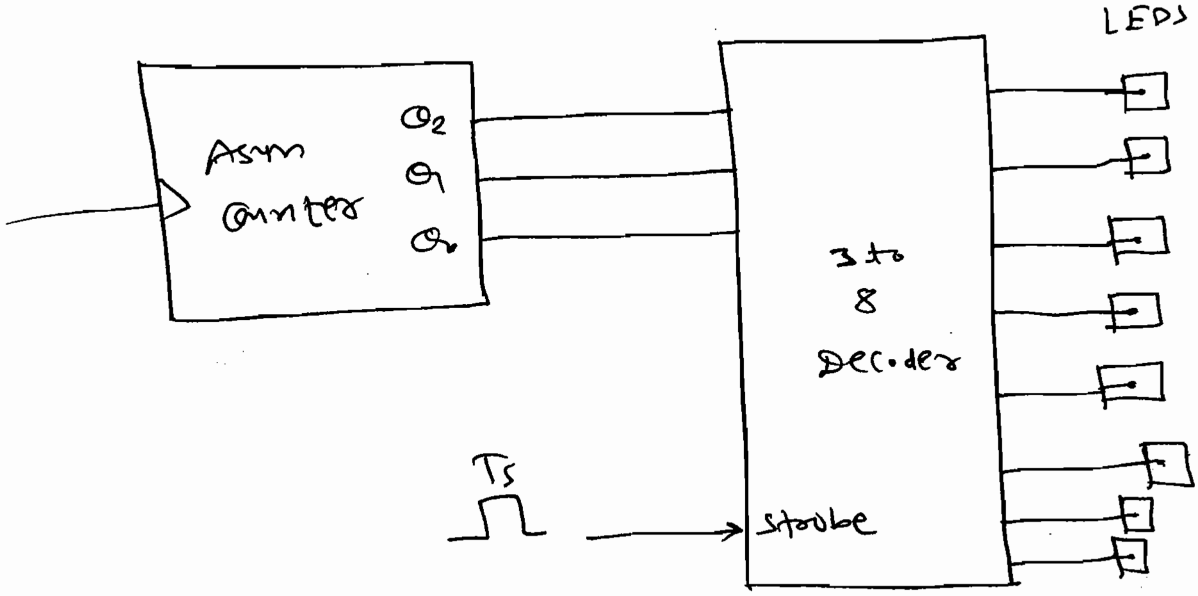
i.e. $f \leq \frac{1}{30 \times 10^{-9}}$.

In general,

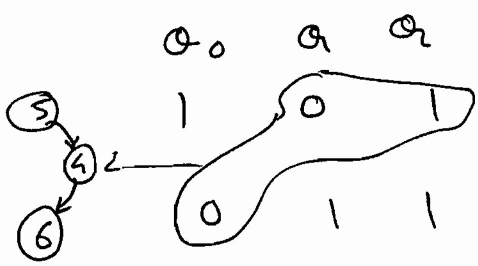
N- Bit Asynchronous Counter,

$$\therefore \boxed{f \leq \frac{1}{N \cdot t_{pd,FF}}}.$$

→ Freq. of Async. Counter using Strobe pulse.



~~Not~~



$$\therefore \boxed{f \leq \frac{1}{N \cdot t_{pd,FF} + T_S}}.$$

→ Asynchronous Counter uses Strobe where as Strobe are not required in Synchronous.

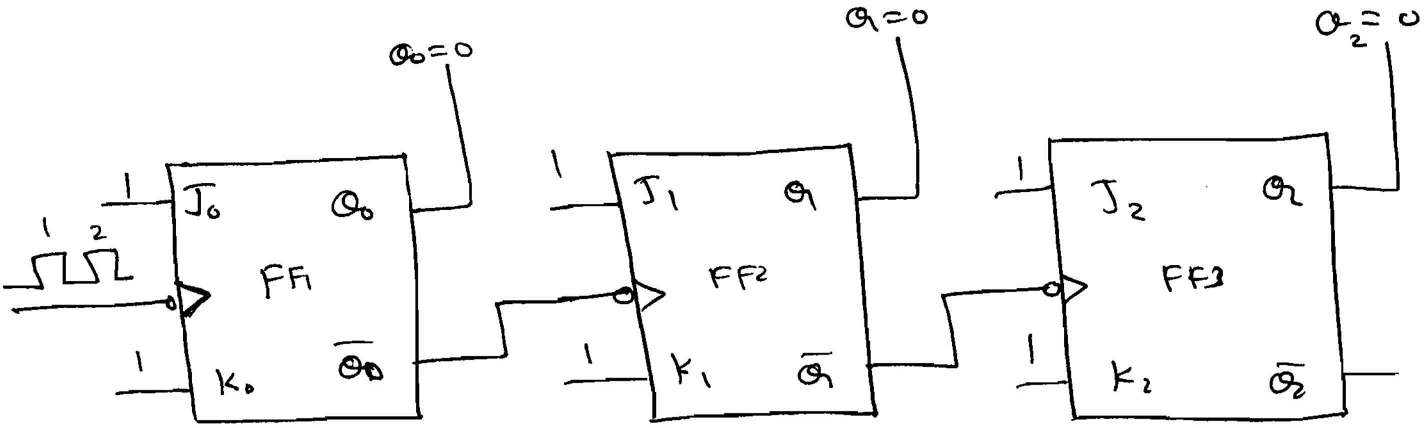
* 3-bit asynchronous counter. (Down Counter).

→ In the following circuits,

(1) Q_0 changes (toggles) for each clock pulse.

(2) Q_1 toggles when Q_0 changes from 0 to 1.

(3) Q_2 toggles when changes from 0 to 1.

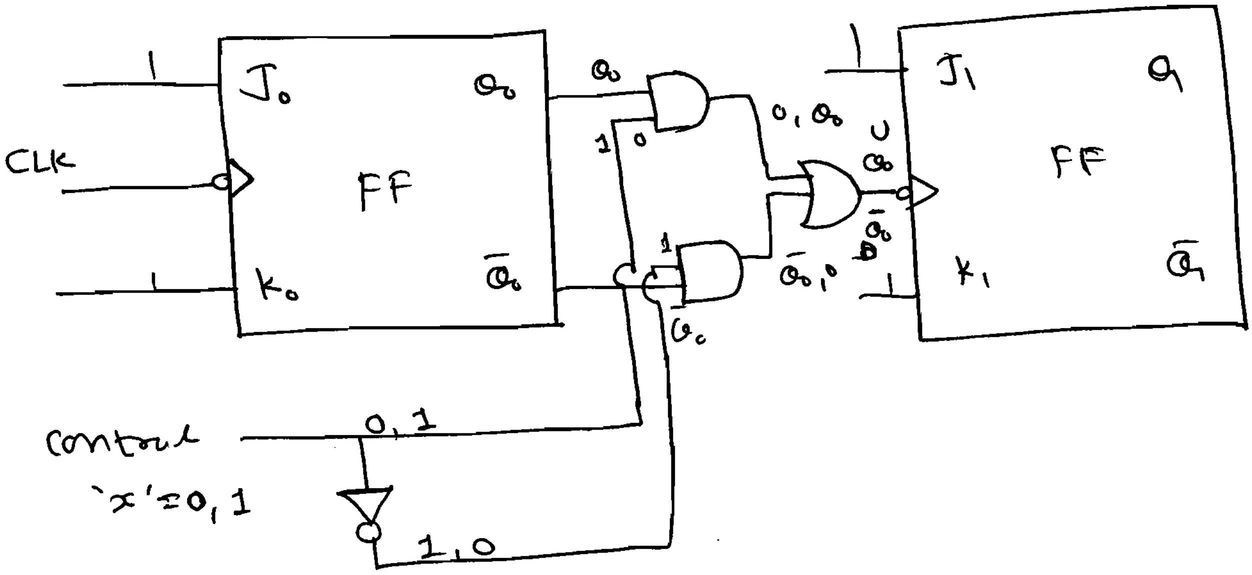


Note: All FFs are in toggle mode.

CLK	Q_0 (LSB)	Q_1	Q_2 (MSB)	Decimal
0	0	0	0	0
1	1	1	1	7
2	0	1	1	6
3	1	0	1	5
4	0	0	1	4
5	1	1	0	3
6	0	1	0	2
7	1	0	0	1
8	0	0	0	0

→ Down Counter.

* 2-Bit "Asynchronous Up-Down" Counter.



→ If $\boxed{x=0}$ ⇒ "$\bar{Q}_0$" is connected to the clock of Next FF.

⇒ $\boxed{\text{Down Counter}}$

→ ~~If~~ ~~x=1~~

i.e. 00, 11, 10, 01, 00, ...

→ If $\boxed{x=1}$ ⇒ "Q_0" is connected to the clock of Next FF.

⇒ $\boxed{\text{Up Counter}}$

i.e. 00, 01, 10, 11, 00, ...

Imp
NOTE: In the above Up-Down Counter if +ve edge triggered FF circuits then

① 'x' = 0 → It act as Up counter.

② 'x' = 1 → It act as Down Counter.

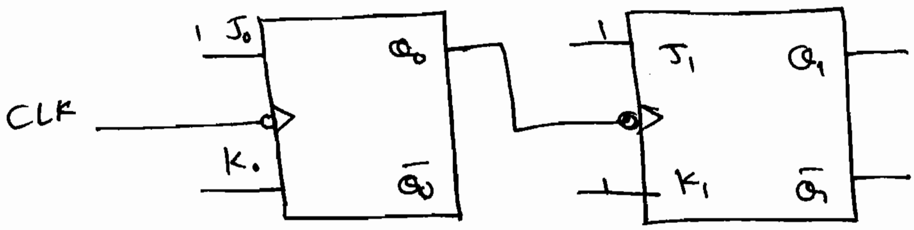
Hint: ✱ $\boxed{\text{Up Counter}}$

triggering	Q
neg. →	positive (or)
pos. →	negative.

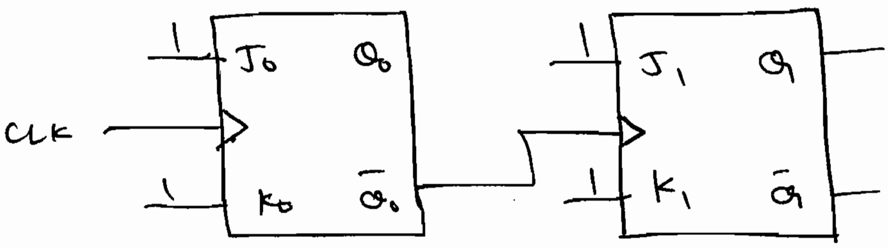
✱ $\boxed{\text{Down Counter}}$

triggering	Q
neg. →	negative
pos. →	positive

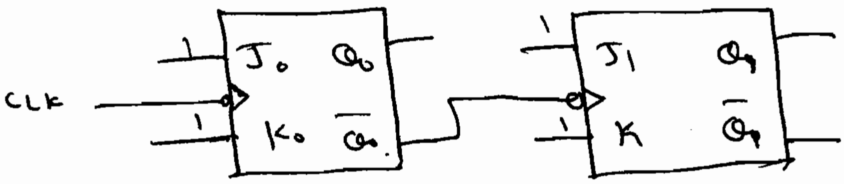
① Up Counter:



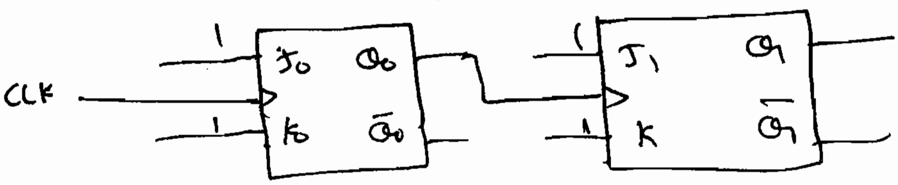
(OR)



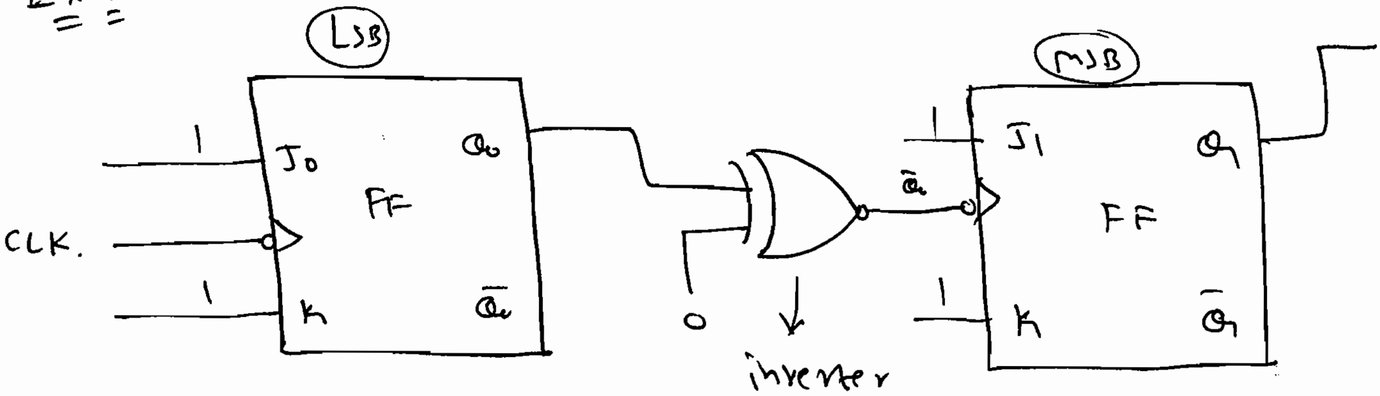
② Down Counter:



(OR)



Ex-1



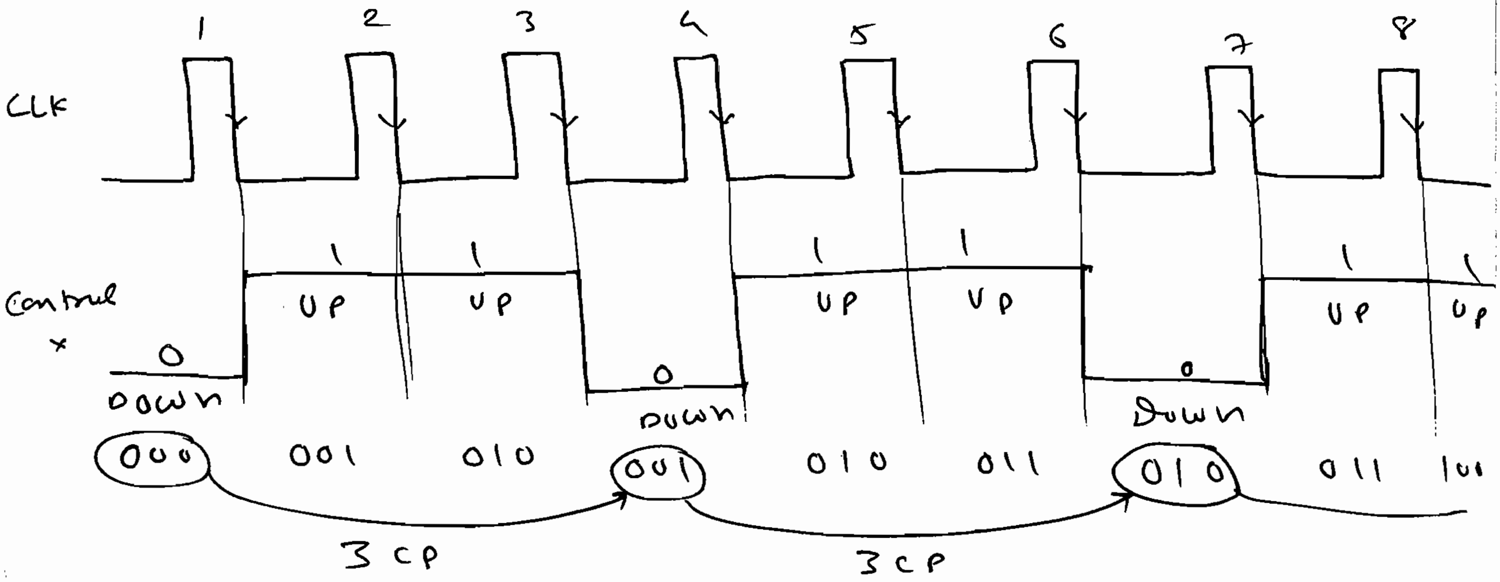
Ans

→ FF is -ve edge triggered, $\bar{Q}_0$ → CLK

hence it is $\boxed{\text{Down Counter.}}$

Imp

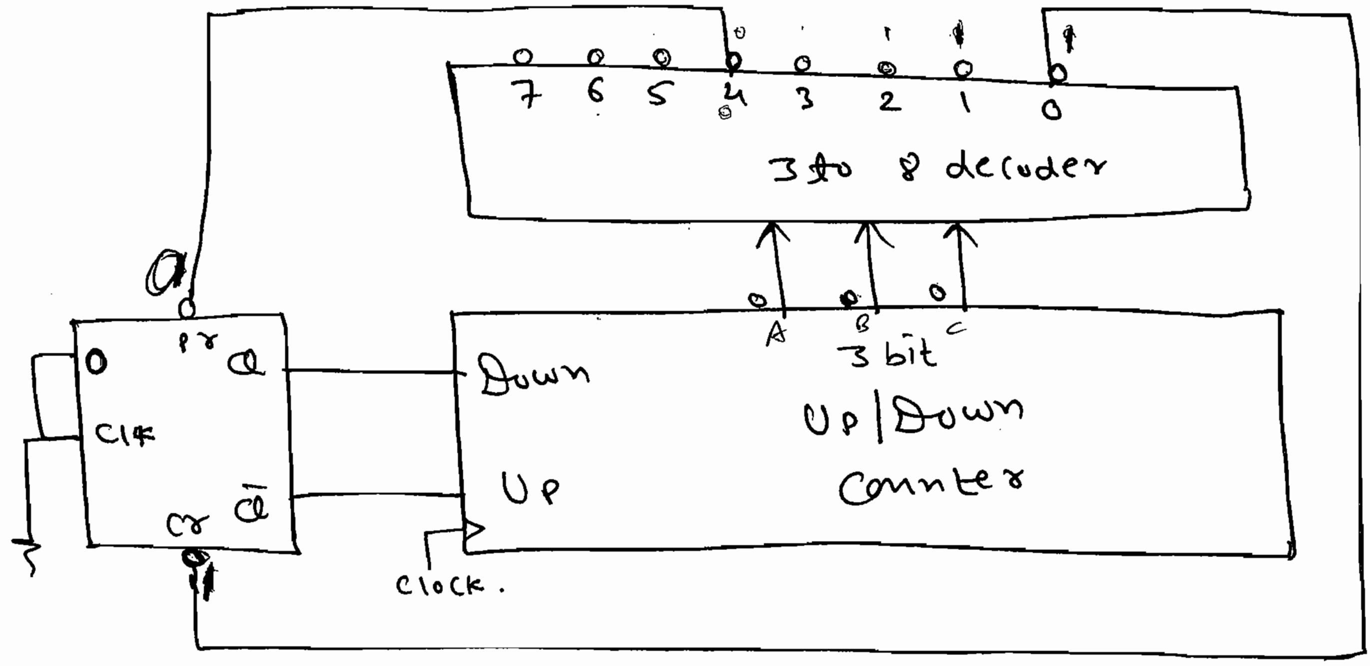
Ex-1 In a 3-bit asynchronous up-Down Counter the clock and control inputs are as given below: Determine the no. of clock pulses required to bring the counter to the initial counting seq. starting with 000.



→ In this Counter each increment requires three clock pulses.

→ To increment 8 times No. of clock pulses are $8 \times 3 = 24$.

Ex-2 In the following Up down Counter determine the counting sequences of the counter.



Ans:

CLK	A	B	C	
0	0	0	0	
				UP
1	0	0	1	
				UP
2	0	1	0	
				UP
3	0	1	1	
				UP
4	1	0	0	
5	0	1	1	Down
6	0	1	0	Down
7	0	0	1	Down
8	0	0	0	Down

$\underbrace{0, 1, 2, 3, 4,}_{\text{UP}}$ $\underbrace{3, 2, 1, 0,}_{\text{DOWN}}$ ....

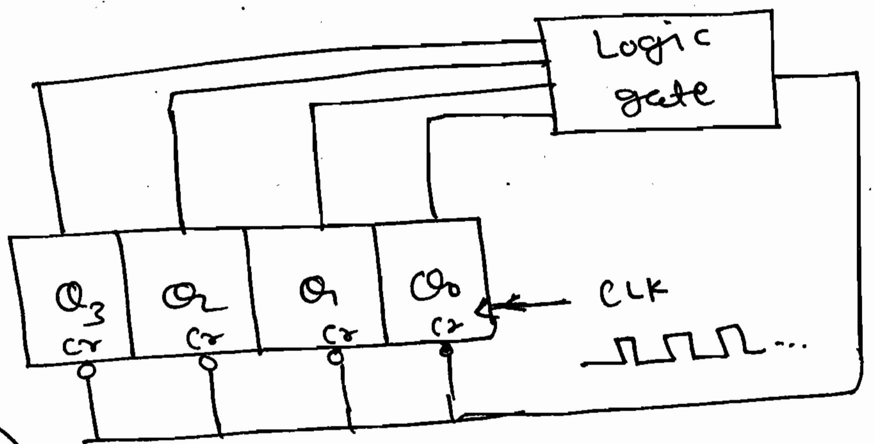
[$\because$ Pr = 1, Cr = 0 i.e. Q = 0, $\bar{Q}$ = 1] [$\because$ Pr = 0, Cr = 1 i.e. Q = 1, $\bar{Q}$ = 0].

☆ Modulus of a counter.

→ It is the number of CLK pulses required to bring the counter in the initial state.

→ A Mod-N counter ~~comes~~ counts from 0 to N-1 and output freq. is f/N.

E.g.



	Q_3	Q_2	Q_1	Q_0
	0	0	0	0
	0	0	0	1
	0	0	1	0
	⋮			
	1	0	0	1
10 →	1	0	1	0
	⋮			
	1	1	1	1

After the required value, F should be zero

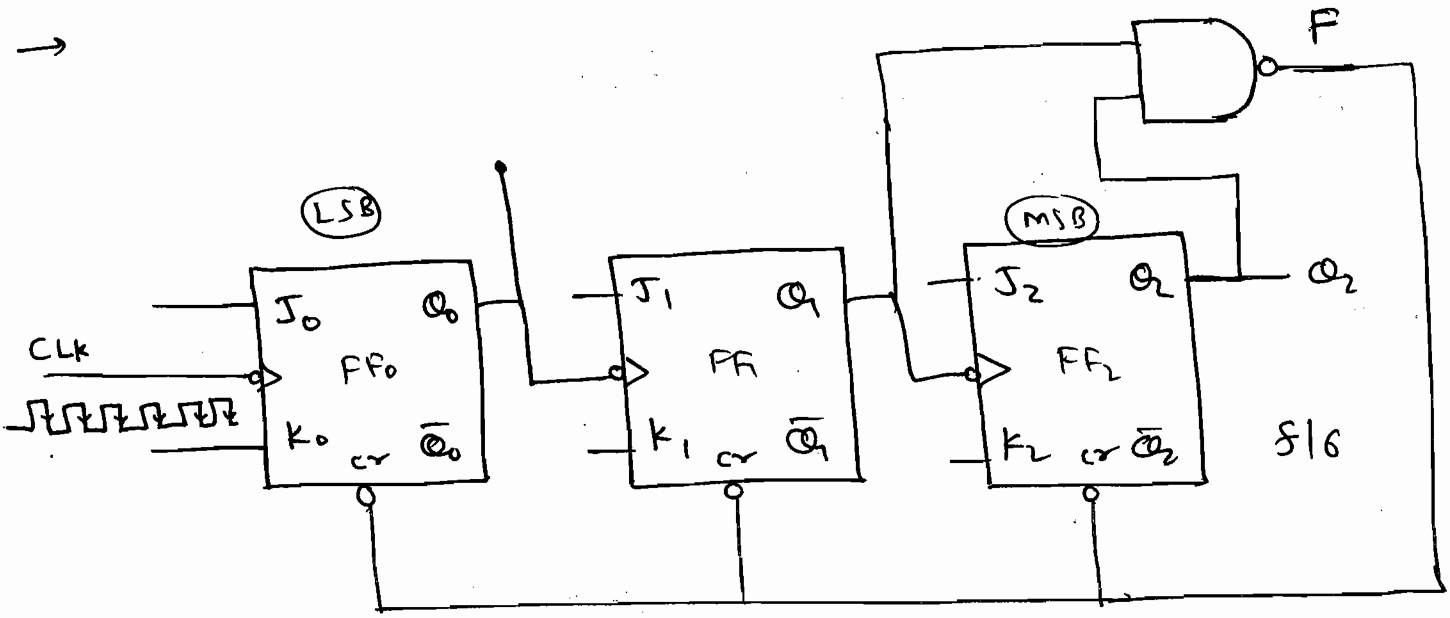
F=0.

i.e for e.g. 12

* Construct a Mod-6 Asynchronous Counter.

→ As soon as it reaches 6 CLK pulse it should be reset.

→



→ 0 to 5 ⇒ $2^N \geq$ No. of mode of counter.

∴ $2^N \geq 6$

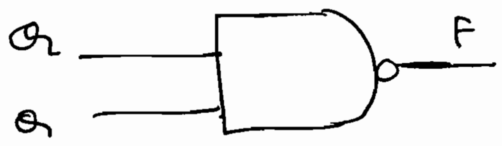
$\boxed{N = 3 \text{ FFs}}$

CLK	Serial inputs	Q_2	Q_1	Q_0	Decimal
0	—	0	0	0	0
1	1	0	0	1	1
2	1	0	1	0	2
3	1	0	1	1	3
4	0	1	0	0	4
5	0	1	0	1	5
6	0	~~1~~	~~1~~	~~0~~	6
		↓ 0	↓ 0	↓ 0	

(i) At 6^{th} CLK pulse $\boxed{Q_2 = 1,\ Q_1 = 1}$

(ii) Hence when $\boxed{Q_2 = 1,\ Q_1 = 1} \Rightarrow$ o/p of gate $\boxed{F = 0.}$ All FFs are cleared.

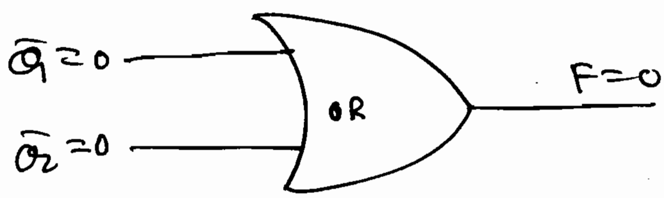
$\Rightarrow$ $\boxed{\text{NAND gate}}$



→ In the above mod-6 asynchronous counter determine the feedback logic gate if its Inputs are $\bar{Q_2}$ & $\bar{Q_1}$

→ i.e. $\bar{Q_2} = 0$ and $\bar{Q_1} = 0 \Rightarrow$ o/p of gate $F = 0.$

$\therefore \Rightarrow \boxed{\text{OR gate.}}$



* Disadvantages:

→ In Asynchronous Counters whose modulus is not equal to 2^N the output produces unwanted spikes called as glitches.

E.g.

	Q_0	Q_1	Q_2
(5) →	1	0	1
(6) →	0	0	0

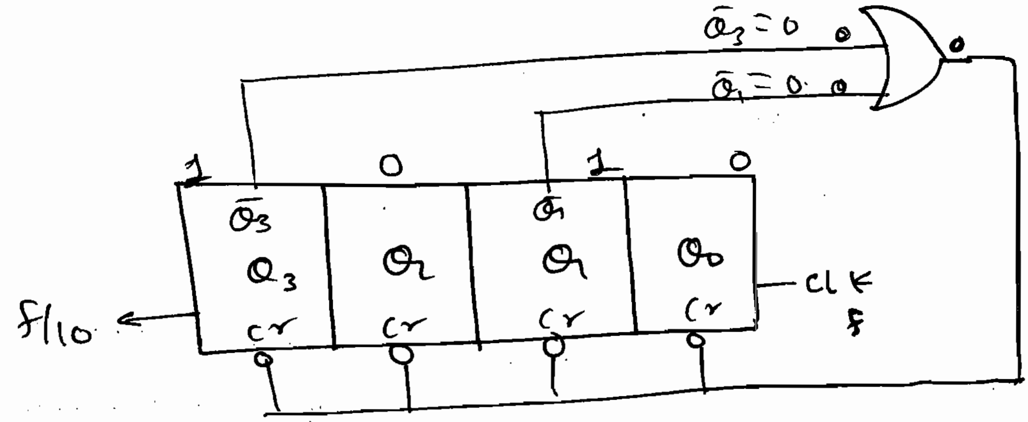
→ **Latch**: output is latched on input, output doesn't change even if input changes

$Q_1 \Rightarrow$  → Glitches.

(2) If the FF are having unequal clearing times then all the FFs cannot be cleared at the required clock pulse. To overcome this a latch is used in the feedback path such that it output remains zero untill all the FFs are cleared.

Ex-1 Determine the modulus of the following Asynchronous Counters:

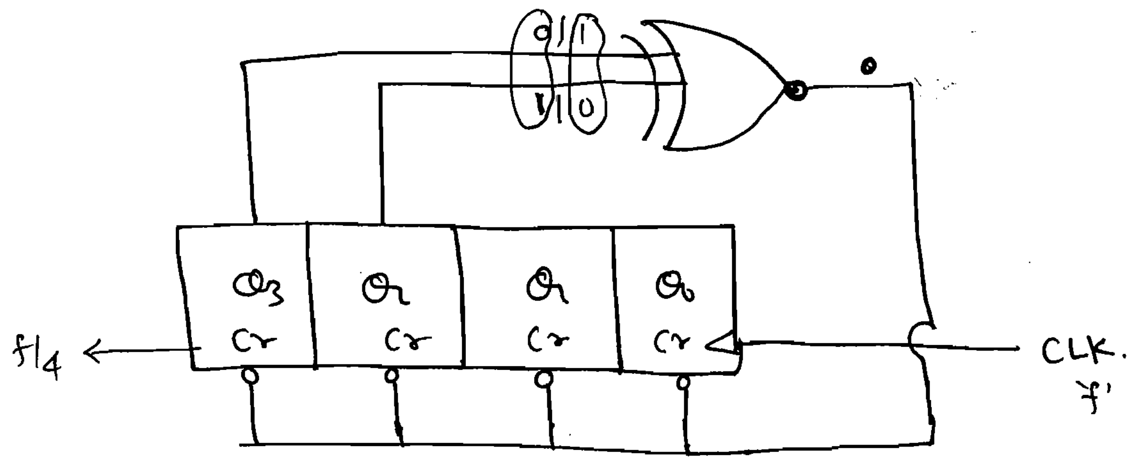
a)



→ For $Q_3Q_2Q_1Q_0 = 1010$ → All the FFs are cleared.

→ It is Mod-10 Counter (Decade Counter)

(b)



Ans:

$\boxed{f \propto \frac{1}{N}}$

Q_3	Q_2	Q_1	Q_0	
0	0	0	0	
0	0	0	1	
	⋮			
~~0~~	~~1~~	~~0~~	~~0~~	0000
	⋮			
1	0	0	0	
1	1	1	1	

4 will appear first i.e 01 will apear first than 10.

0 1 2 3 ~~4~~

For $Q_3 Q_2 Q_1 Q_0$ = 0100 → All the FFs are cleared

→ It is Mod - 4 Counter.

→ The disadvantages of Asynchronous Counter is the freq. of operation is inversly proportional to the no. of FFs. To overcome this we use Synchronous or parallel Counters.

* Flip Flop Excitation table:

① S-R FF:

Q(t)	Q(t+1)	S	R
0 →	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

② D- Flip Flop:

Q (t)	Q (t+1).	D
0	0	0
0	1	1
1	0	0
1	1	1

③ T- Flip Flop.

Q(t)	Q(t+1)	T
0	0	0
0	1	1
1	0	1
1	1	0

④ J-K- Flip flop.

Q(t)	Q(t+1)	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

* Summary of FF excitation table:

		FF Inputs					
Q(t)	Q(t+1)	J	K	S	R	D	T
0	0	0	X	0	X	0	0
0	1	1	X	1	0	1	1
1	0	X	1	0	1	0	1
1	1	X	0	X	0.	1	0

Ex-1 Determine the exitation table of X-Y FF whose T.T. is as given below.

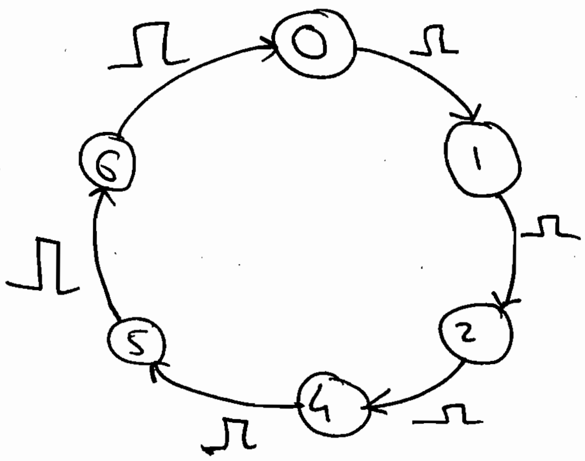
X	Y	Q(t+1)
0	0	1
0	1	Q(t)
1	0	$\bar{Q}(t)$
1	1	0

Q(t)	Q(t+1)	X	Y
0	0	X	1
0	1	X	0
1	0	1	X
1	1	0	X

* Design a Synchronous Counter using J-K Flip Flops. which Comes through the states 0, 1, 2, 4, 5, 6, 0, ...

(b) Is it a self starting Counter?

Ans: (a) State Diagram.



(b) State Assignment:

0 → 000

1 → 001

2 → 010

4 → 100

5 → 101

6 → 110.

(c) Excitation Table:

	Q_2	Q_1	Q_0	$J_2 K_2$	$J_1 K_1$	$J_0 K_0$
				FF Inputs		
0	0	0	0	0 X	0 X	1 X
1	0	0	1	0 X	1 X	X 1
2	0	1	0	1 X	X 1	0 X
4	1	0	0	X 0	0 X	1 X
5	1	0	1	X 0	~~0 X~~ 1 X	X 1
6	1	1	0	X 1	X 1	0 X
7	0	0	0			

* 3, 7 are unused states. Takes them as dont cares.

$J_0 = \bar{Q_1}$ $\quad K_0 = 1$

$J_1 = Q_1 + Q_2$ $\quad K_1 = 1$

$J_2 = Q_1$ $\quad K_2 = Q_1$

$Q_0 \backslash Q_1 Q_2$	00	01	11	10
0	0	0	X	1
0	X	X	X	X

$J_2 = Q_1$.

$Q_0 \backslash Q_1 Q_2$	00	01	11	10
0	X	X	X	X
1			X	1

$K_2 = Q_1$

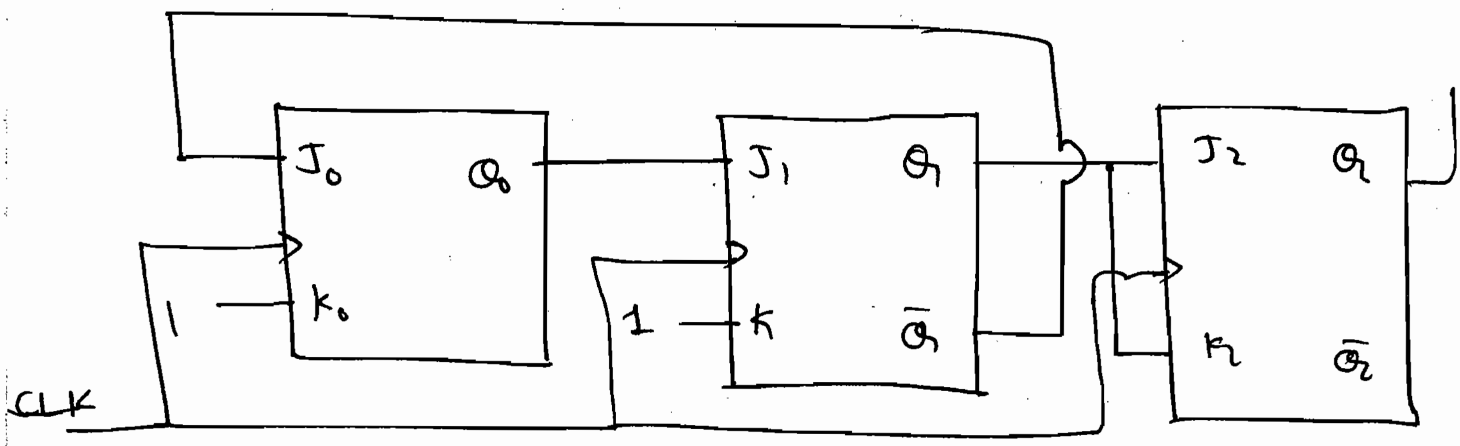
$Q_0 \backslash Q_1 Q_2$	00	01	11	10
0	1	X	X	
1	1	X	X	

$J_0 = \bar{Q_1}$

$Q_0 \backslash Q_1 Q_2$	00	01	11	10
0		1	X	X
1		1	X	X

$J_1 = Q_2$ ~~+ Q~~.

Ex/2

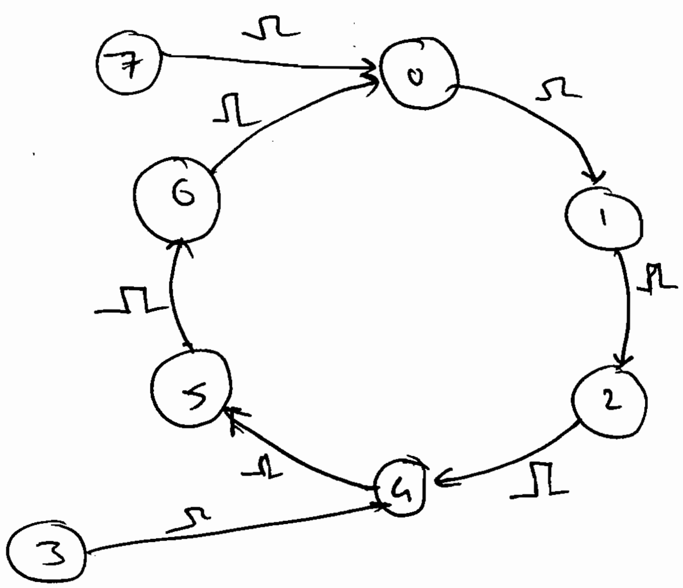


→ In Sync Counter, Freq of operation

$$\boxed{f \le \frac{1}{t_{pd.FF}}}$$

(b)

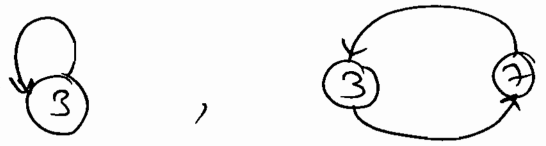
	Present State			FF Inputs			Next State		
	Q_2	Q_1	Q_0	J_2K_2	J_1K_1	J_0K_0	Q_2	Q_1	Q_0
3 →	0	1	1	1 1	1 1	0 1	1	0	0 (4)
7 →	1	1	1	1 1	1 1	0 1	0	0	0 (0)



It is a self starting counter as we get (4) after (3) and (0) after (7)

→ The above counter is a self starting counter because it is able to enter the used states from all unused state.

e.g. of non self starting counter



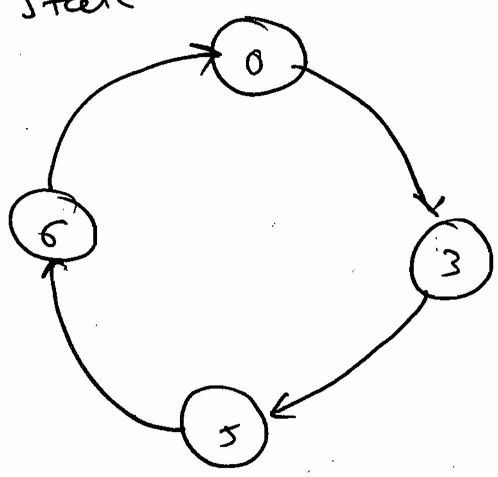
Ex-3

A synchronous counter comes through the states 0, 3, 5, 6, 0, ... and FF inputs are $T_2 = Q_1$, $T_1 = 1$, $T_0 = \bar{Q}_1$. Is it a self starting counter.

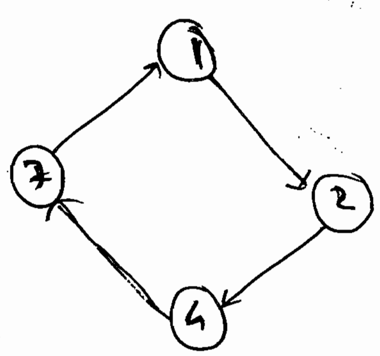
⇒ Table for analysis:

	Pr. State Q_2	Q_1	Q_0	FF Inputs $T_2 = Q_1$	$T_1 = 1$	$T_0 = \bar{Q}_1$	Next State Q_2	Q_1	Q_0	
1	0	0	1	0	1	1	0	1	0	2
2	0	1	0	1	1	0	1	0	0	4
4	1	0	0	0	1	1	1	1	1	7
7	1	1	1	1	1	0	0	0	1	1

Used state



Unused state



↓

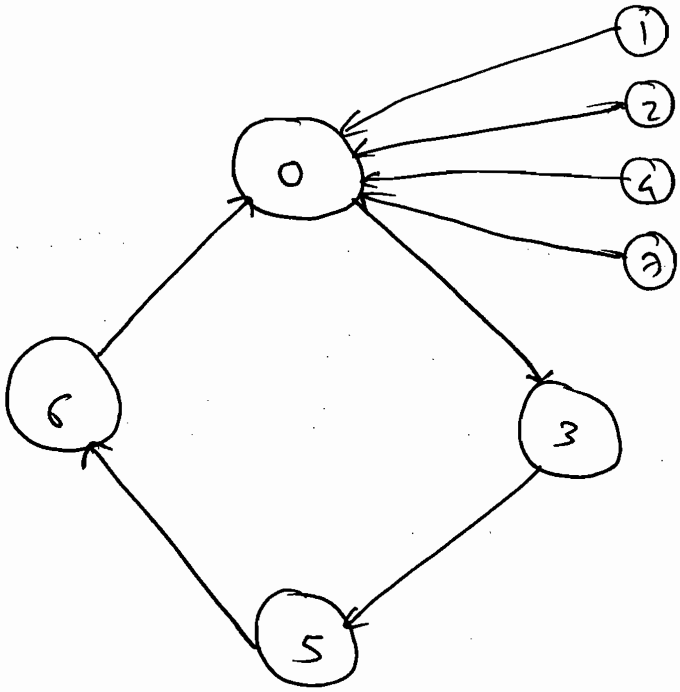
'Lock-out' of counter

Not a self starting counter

→ It is an efficient counter.

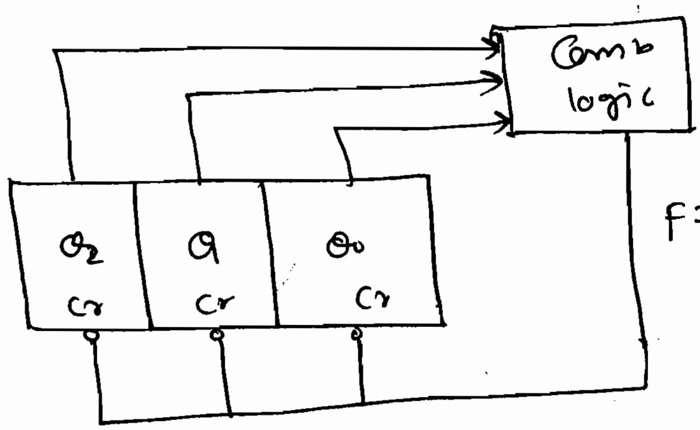
→ To avoid the Lock-out condition

① Redesign the counter according to the following state diagram.



→ No. of FF state is increasing ⇒ no. of FF is increases in it will increase the complexity of ckt

② Clearing the counter as soon as it enters into unsed state. as shown below:



F = 0/1 if the o/B

$F(Q_2, Q_1, Q_0) = \Sigma m(1, 2, 4, 7)$.

Q_2 \ Q_1Q_0	00	01	11	10
0		1		1
1	1		1	

$$\therefore \quad F = Q_2 \oplus Q \oplus Q_0$$

(or)

$$F = Q_2 \odot Q \odot Q_0$$

NOTE:

⇒ In the above synchronous counter the combinational logic is determine as follows:

→ When counter enters into the states ①, ②, ④, ⑦ the F should become 1. So, that the counter can be clear.

→ The sum of minterm expression for F is

$$F(Q_2, Q_1, Q_0) = \Sigma(1, 2, 4, 7).$$

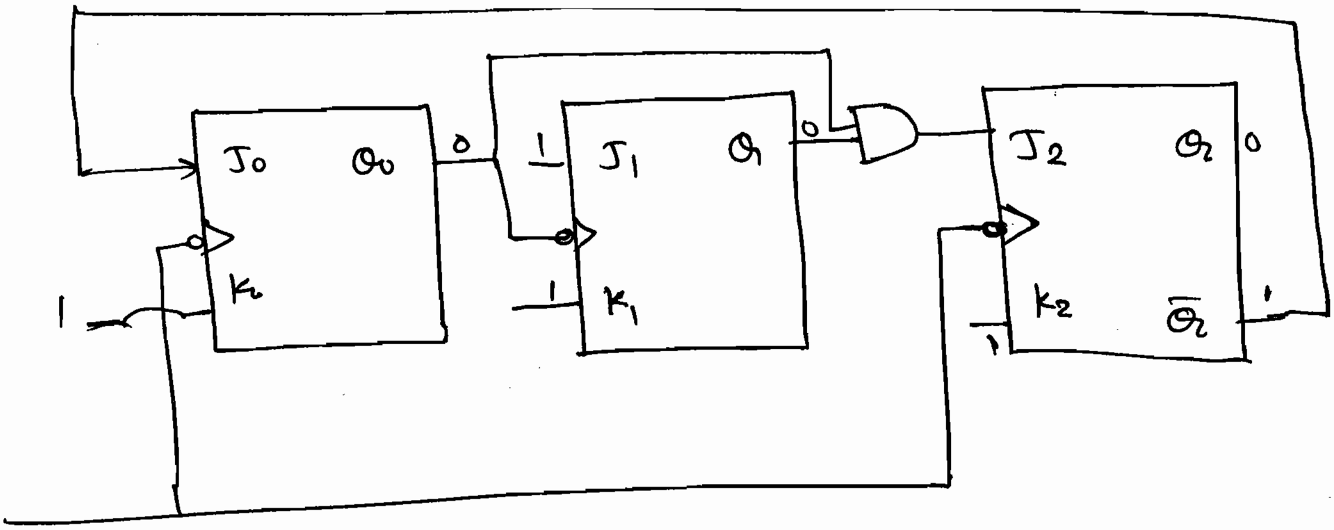
Q_2 \ Q_1Q_0	00	01	11	10
0		1		1
1	1		4	

$$\therefore \quad F = Q_1 \oplus Q_2 \oplus Q_3$$

(or)

$$F = Q_1 \odot Q_2 \odot Q_3.$$

Ex-2 Determine the modulus of the following counter

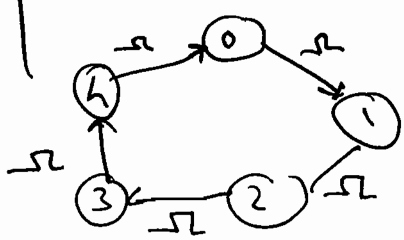


Ans: 'Q_1' is in Asynchronous mode. It toggle when Q_0 changes from '1 to 0'.

Now
$J_0 = \bar{Q}_2$, $\quad J_2 = Q_1 Q_0$
$K_0 = 1$ $\quad K_2 = 1$.

⇒ Table for analysis:

P.S.			FF Inputs				N.S.			
Q_2	Q_1	Q_0	J_2	K_2	J_0	K_0	Q_2	Q_1^*	Q_0	
⓪ 0	0	0	0	1	1	1	0	0	1*	①
① 0	0	1	0	1	1	1	0	1	0	②
② 0	1	0	0	1	1	1	0	1	1	③
③ 0	1	1	1	1	1	1	1	0	0	④
④ 1	0	0	0	1	0	1	0	0	0	⓪



"Modulus = 5".

$\rightarrow$ If $t_{pd,FF} = 10ns \rightarrow f_{max} = ?$

Max. conv. time $= 10 + 10 = 20ns$.

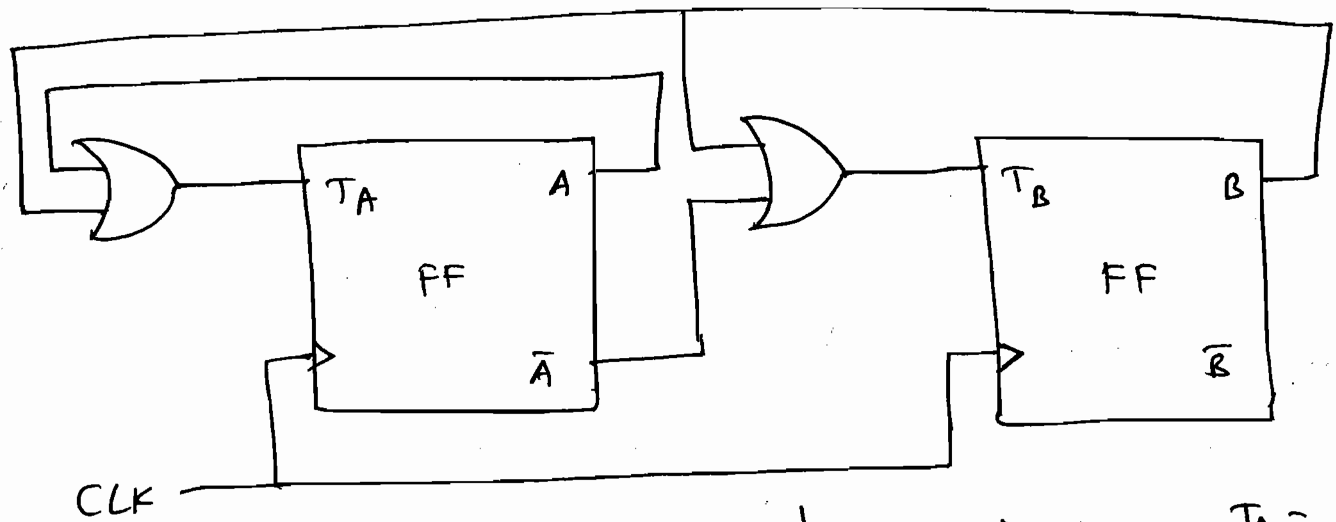
[$\because$ FF_0, FF_2 are in synch. mode,
FF_1 is in Async mode].

Clock period $\boxed{T \geq 20ns.}$

$\therefore f \leq \frac{1}{20 \times 10^{-9}}$

$\therefore \boxed{f_{max} = \frac{1}{20 \times 10^{-9}} \ Hz}$

Ex-3 Determine the f^n of the following seq. circuits by obtaining its state diagram.

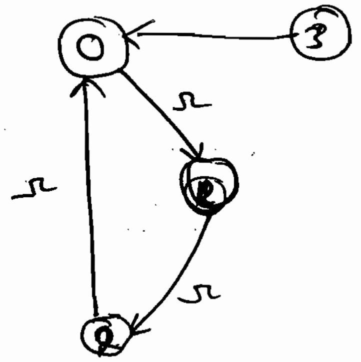


$T_A = A + B$

$T_B = \bar{A} + B$

	P.S.		FF inputs		N.S.		
	A	B	T_A	T_B	A	B	
0	0	0	0	1	0	1	1
1	0	1	1	1	1	0	2
2	1	0	1	0	0	0	0
3	1	1	1	1	0	0	0

State diagram:



→ it is a MOD-3, self starting synchronous counter.

* State Diagram:

Hint:

(i) No of states = 2^N ; N = No of FFs

(ii) No of branches from each state = 2^x ; x = no of direct inputs.

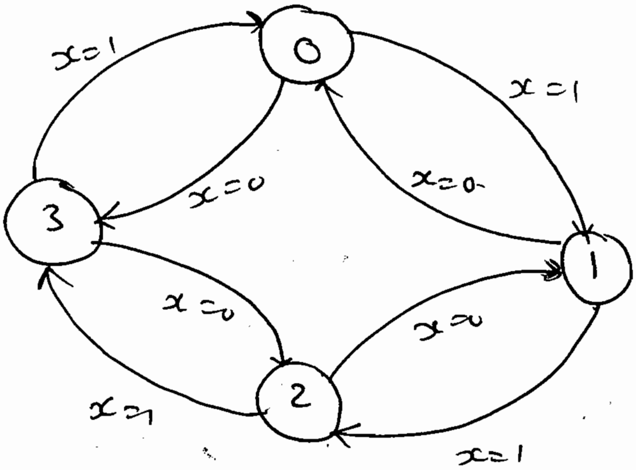
① 2-Bit Up/Down sync Counter:

→ 2FFs ⇒ No. of states = $2^2 = 4$.

($Q_1 Q_0$ = 00, 01, 10, 11)

00 → 0, 01 → 1, 10 → 2, 11 → 3

→ 1 direct i/p (i.e. x) ⇒ No. of branches from each state = $2^1 = 2$.

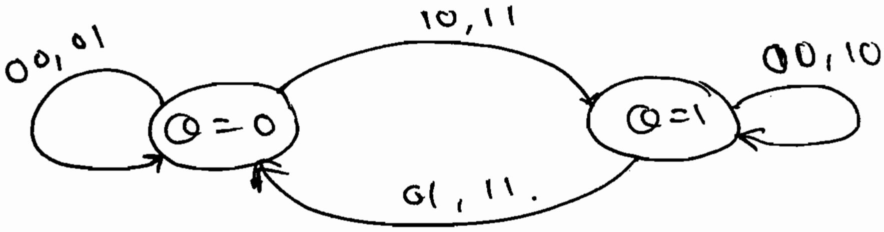


[State Diagram]

② J-K FF state diagram:

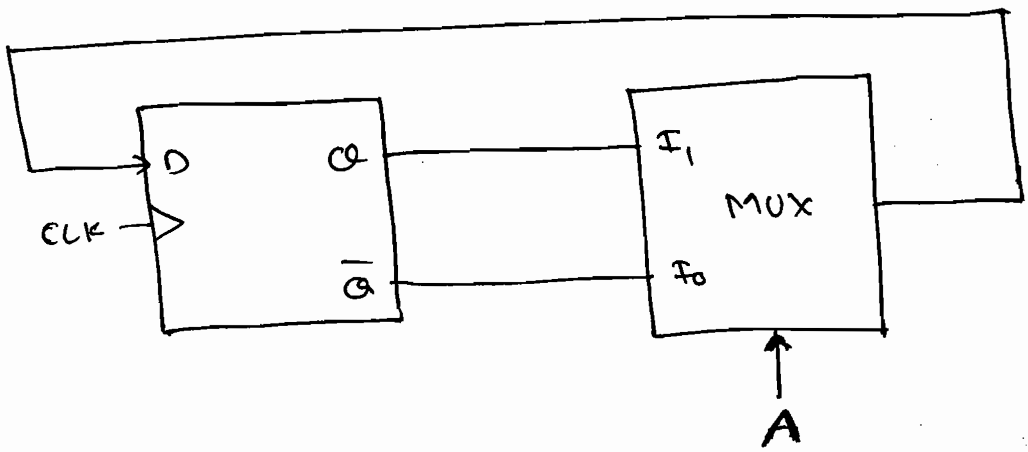
→ 1 FF ⇒ 2 states (Q = 0, Q = 1).

→ 2 Direct input ⇒ 4 branches from each state.



Q(t)	J	K	Q(t+1)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

③ Gate - 2013



→ 1 FF ⇒ 2 States (Q=0, Q=1).

→ Input (i.e A) ⇒ 2 branches from each State.

For D-FF → $Q(t+1) = D$ — ①

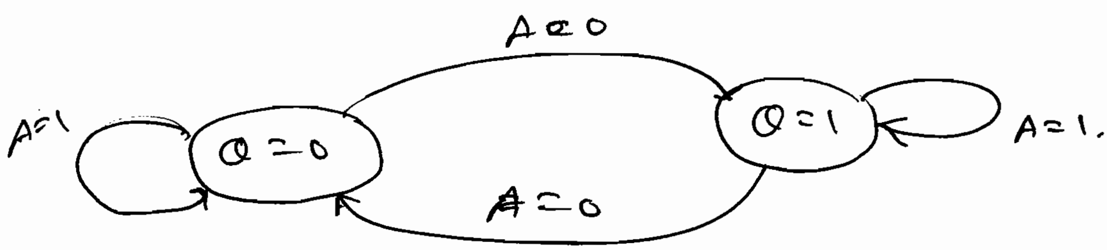
For MUX → $D = \bar{A} \cdot \bar{Q} + AQ$ — ②

from ① & ②

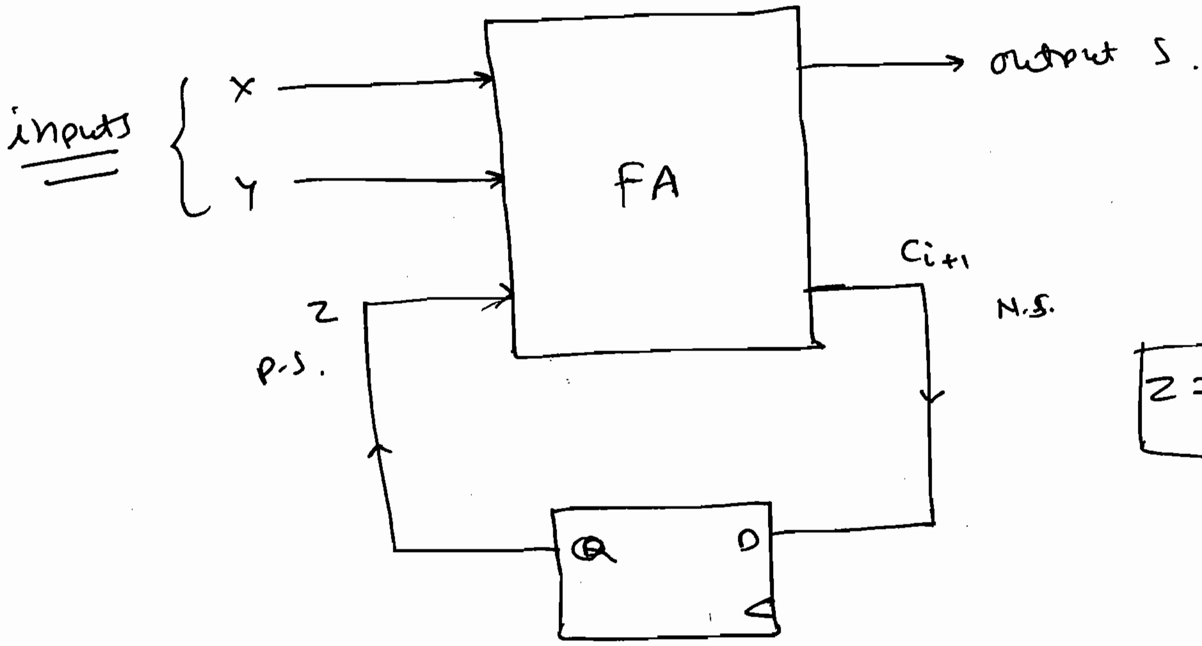
$\therefore \quad Q(t+1) = \bar{A} \cdot \overline{Q(t)} + A \cdot Q(t)$.

When $A = 0$, $Q(t+1) = \overline{Q(t)}$

$A = 1$, $Q(t+1) = Q(t)$.



(4) Serial Adder State diagram:

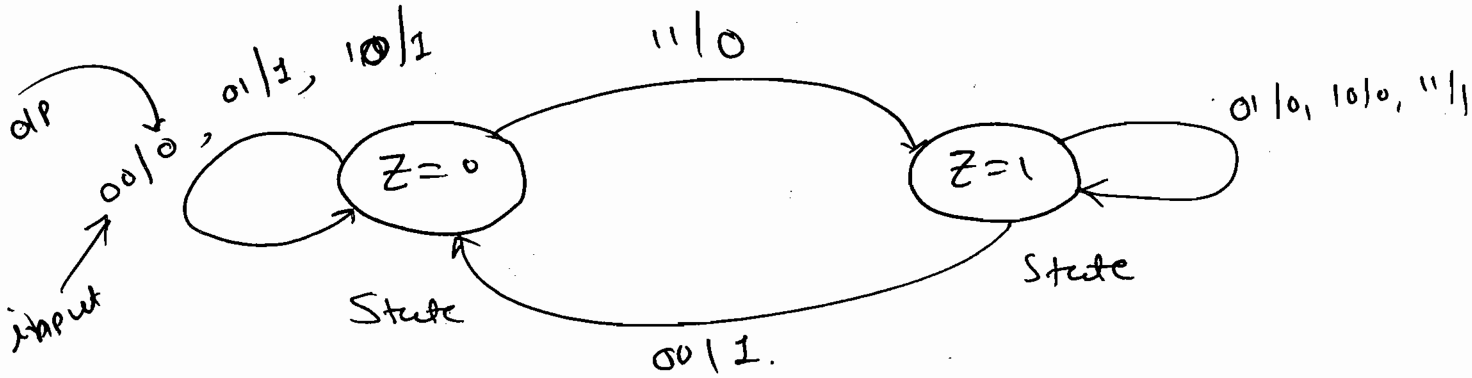


$Z = Q = C_{i+1}$
$= Q(t+1)$

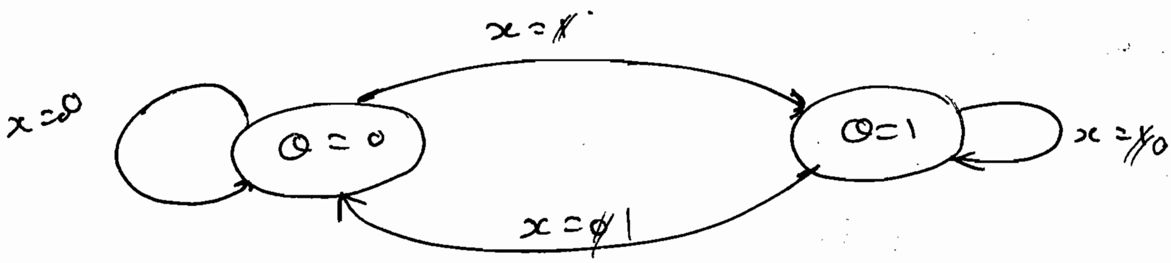
→ ~~2/input~~ 1 FF → 2 states (z=0, z=1).

→ 2 input → $2^2 = 4$ branches from each state.

P.S. (Z)	input X	Y	N.S. C_{i+1}	output Sum
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

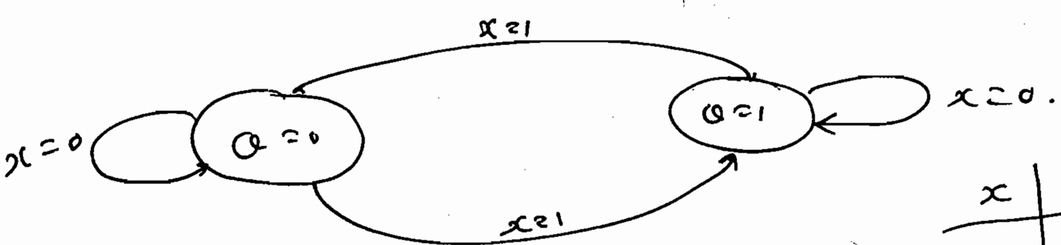


Ex-1 Identify the following FFs.



x	$Q(t+1)$
0	0
1	1

→ T-FF State diagram

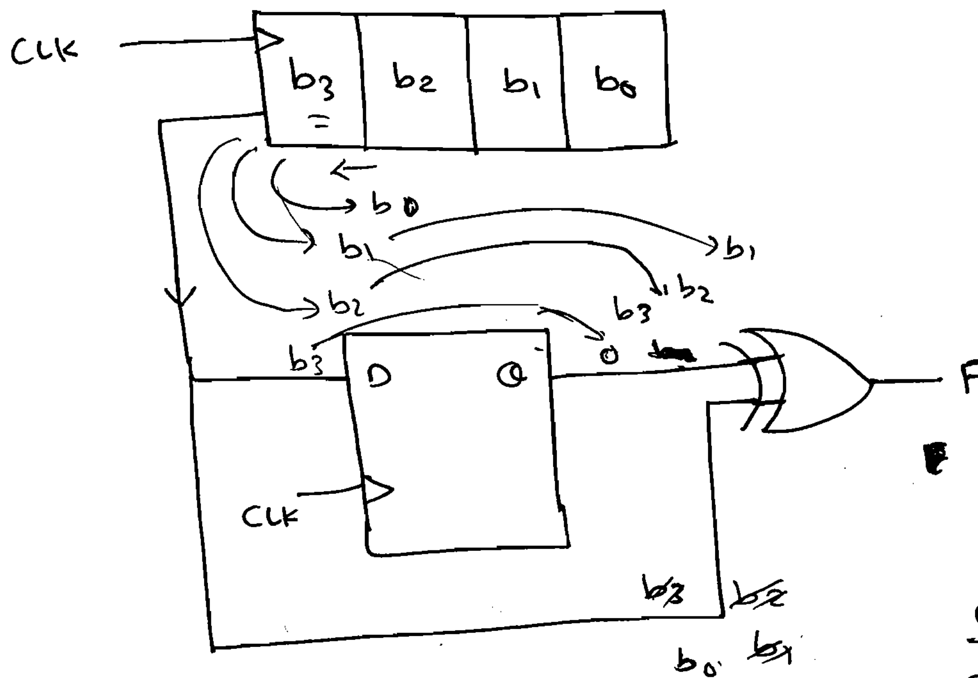


x	$Q(t+1)$
0	$Q(t)$
1	$\bar{Q}(t)$

Ex-1 Determine the f^n of the following circuits.

(1)

Shift left shift Register.



$G_3 = b_3 \oplus 0$.

$G_3 = b_3$.

$G_2 = b_3 \oplus b_2$.

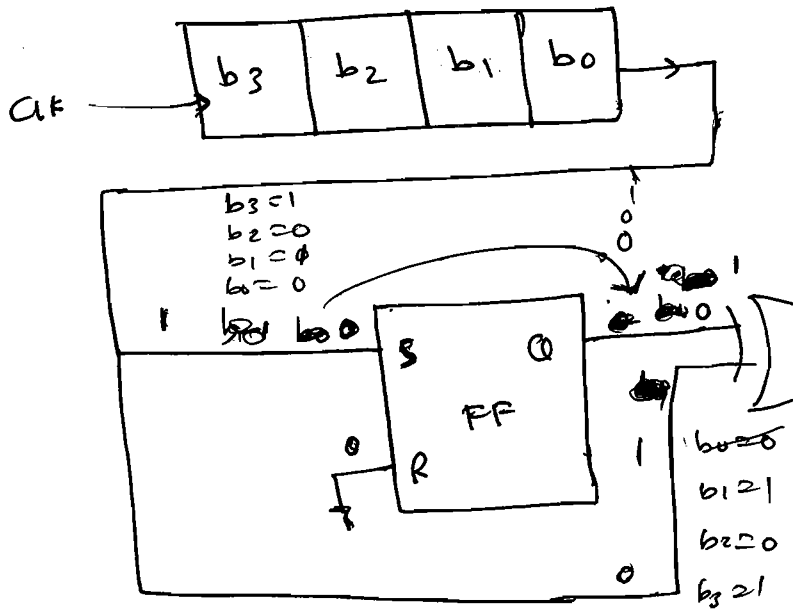
$G_1 = b_2 \oplus b_1$

$G_0 = b_1 \oplus b_0$.

→ $\boxed{\text{Binary to Gray Code Conversion.}}$

(2)

Shift Right Shift Register



1010

0
0
1
0
1 0 1 0
0 1 0 0
0

$b_0 = 0 \oplus 0 = 0$

$b_1 = 1 \oplus 0 = 1$

$b_2 = 0 \oplus 1 = 1$

$b_3 = 1 \oplus 1 = 0$.

Input = $\boxed{1\ 0\ 1\ 0}$

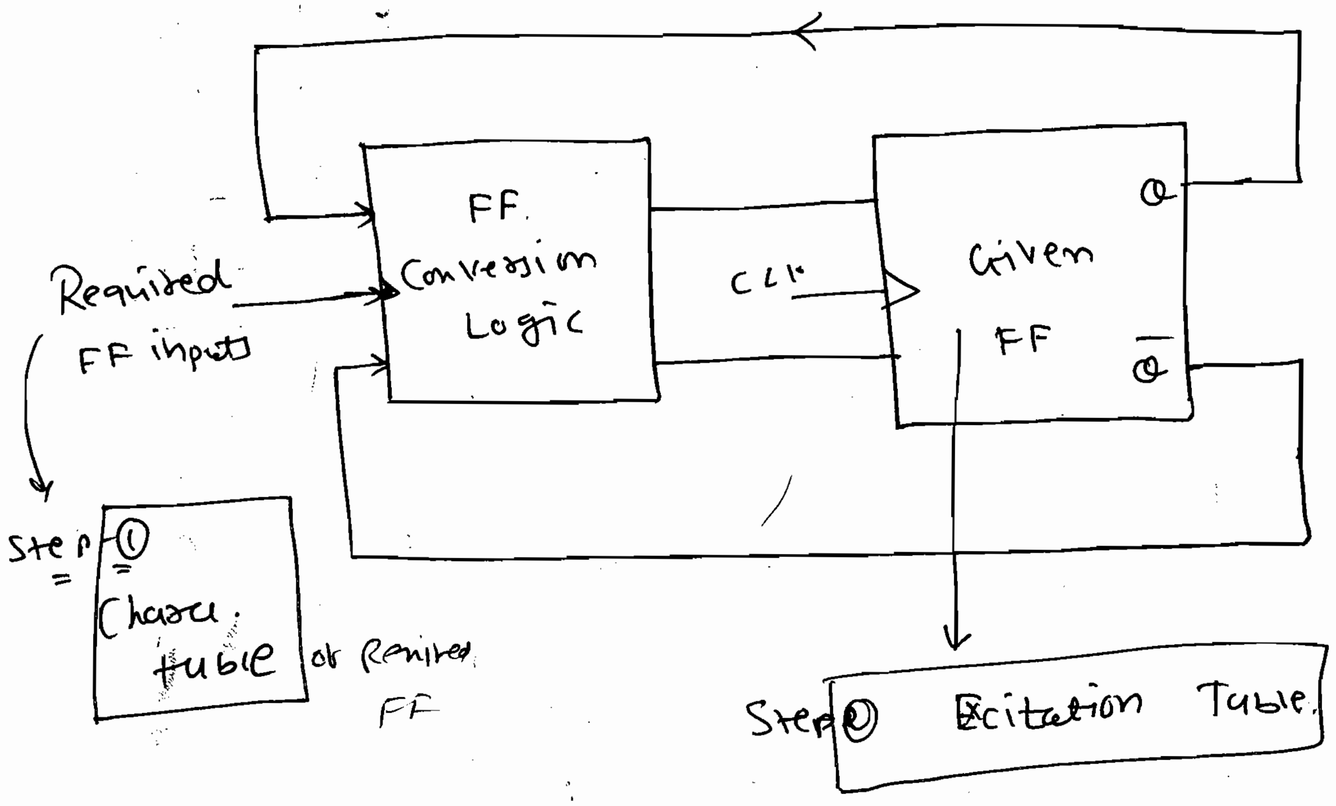
output = $\boxed{0\ 1\ 1\ 0}$

→ It is 2's comp. of binary no.

→ so, $\boxed{\text{it is 2's complement circuit}}$

* Conversion of Flipflops:

⇒ Universal principle:



Ex-1 Convert S-R FF to T flip flop.

Ans: Step-① → Characterestics table of T-Flip flop.

Step-② → Excitation table of SR flip flop.

T	Q(t)	Q(t+1)	S	R
0	0	0	0	x
0	1	1	x	0
1	0	1	1	0
1	1	0	0	1

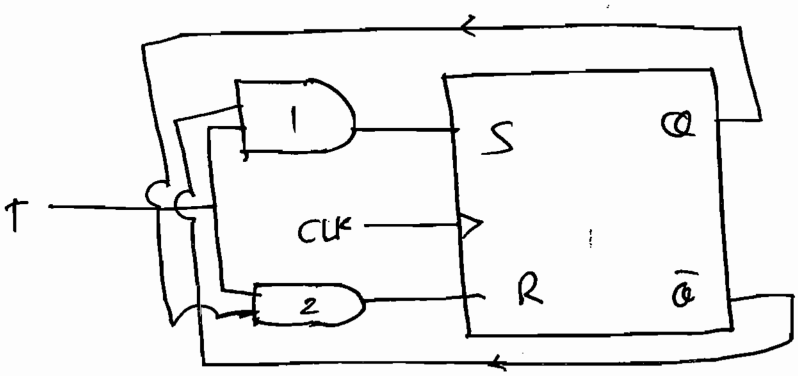
→ (S)

T \ Q(t)	0	1
0	0	x
1	1	0

$S = T\bar{Q}(t).$

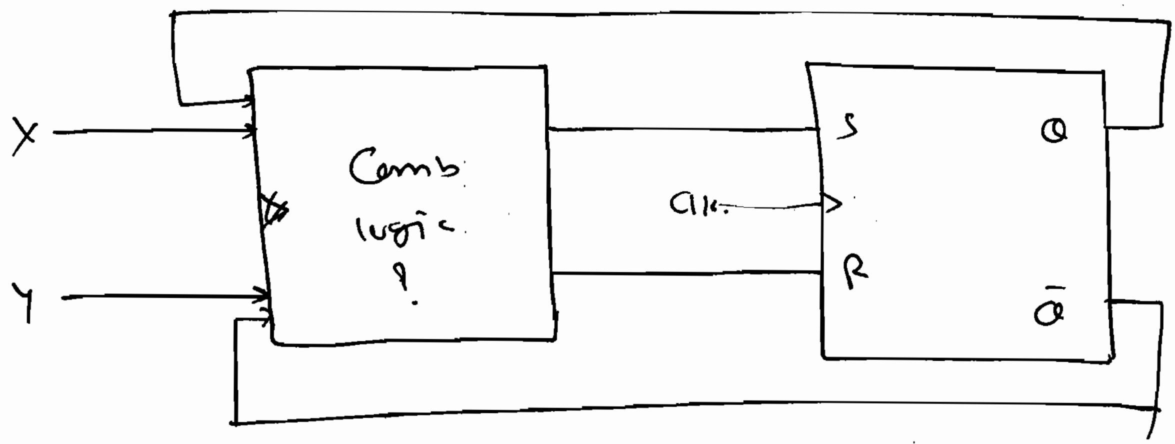
T \ Q(t)	0	1
0	x	0
1	0	1

$R = TQ(t).$



[T- Flip Flop]

Ex-2 In the following diagram determine the combinational logic to be used.



Given ⇒

X Y	Q(t+1)	S R
0 0	0	
0 1	Q(t)	
1 0	$\bar{Q}$(t)	
1 1	1	

X	Y	Q(t)	Q(t+1)	S	R
0	0	0	0	0	X
0	0	1	0	~~X~~ 0	~~0~~ 1
0	1	0	0	0	~~X~~
0	1	1	1	X	0
1	0	0	1	1	0
1	0	1	0	0	1
1	1	0	1	1	0
1	1	1	1	X	0

① S

$x \backslash yQ(t)$	00	01	11	10
0	0	ϕ	X	0X
1	1	0	X	1

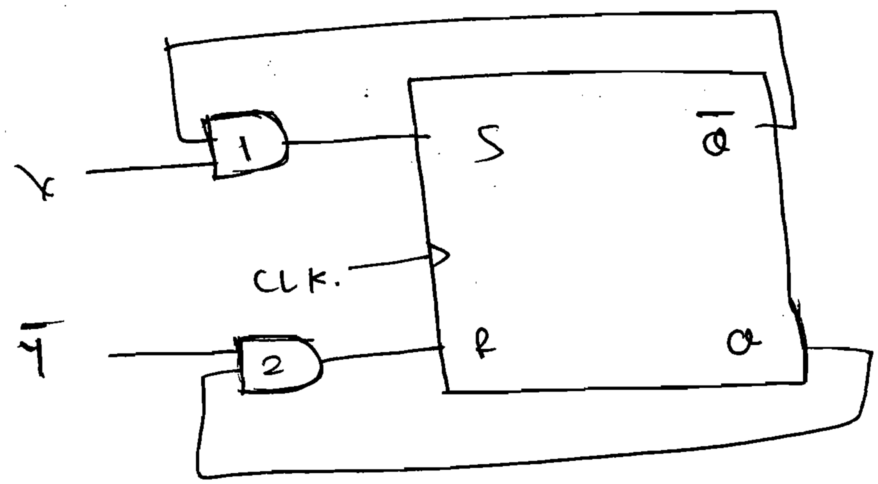
$\therefore \quad S = \cancel{Y}\cancel{A}\, X \cancel{Y}.\overline{Q(t)}$

② R

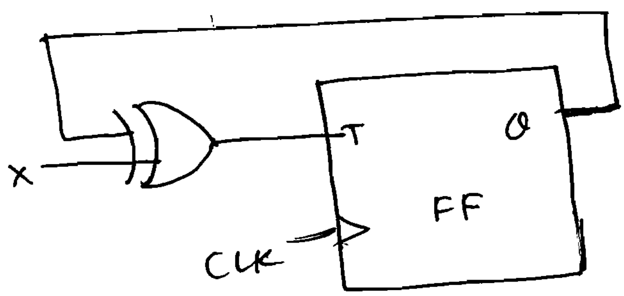
$x \backslash yQ(t)$	00	01	11	10
0	X	ϕ	0	0
1	0	1	0	0

$\therefore \cancel{R = X\bar{Y}Q(t)}$

$R = \bar{Y} Q(t).$



Ex-3 (a) Identify the following flip flop.



$T = x \oplus Q(t) \quad -①.$

For T FF

$T \Rightarrow Q(t+1) = T \oplus Q(t)$

$Q(t+1) = x \oplus Q(t) \oplus Q(t)$

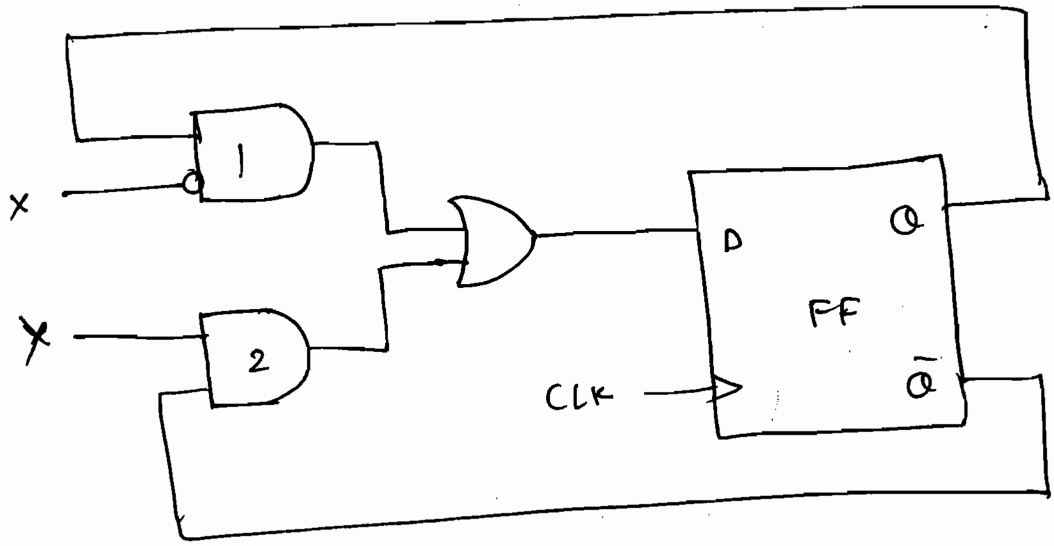
$\therefore Q(t+1) = x \oplus 0$

$\boxed{Q(t+1) = x}$

We know $\Rightarrow$ (3) is similar to D-FF Char eqn,

$\Rightarrow$ It is D-Flip Flop.

b)



$D = Q\bar{X} + \bar{Q}Y.$

$J\bar{Q}$ $\bar{J}Q(t)$

$KQ.$ $K\bar{Q}(t).$

$\therefore Q(t+1) = Q(t)\bar{X} + \bar{Q}(t)Y. \quad \text{—(3)}$

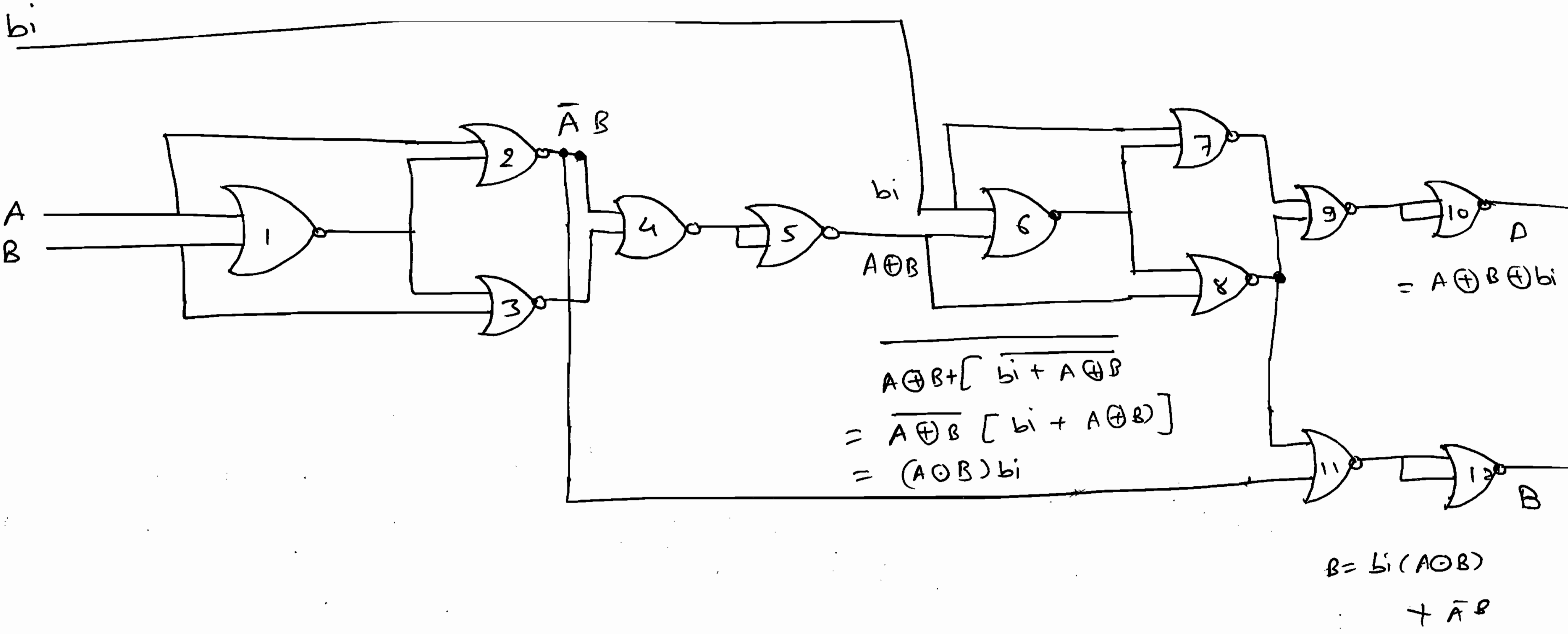
$\therefore Q(t+1) = \bar{J}\,\overline{Q(t)} + \bar{K}Q(t). \quad \text{—(4)}$

So, $\bar{X} = \bar{K}$ $\boxed{Y = J}$ $\boxed{\begin{matrix} X = K \\ Y = J \end{matrix}}$

$\boxed{\cancel{X = J}}$ $\boxed{X = K}$

So, it is J-K flip flop.

* Full Subtractor using NOR gate only.



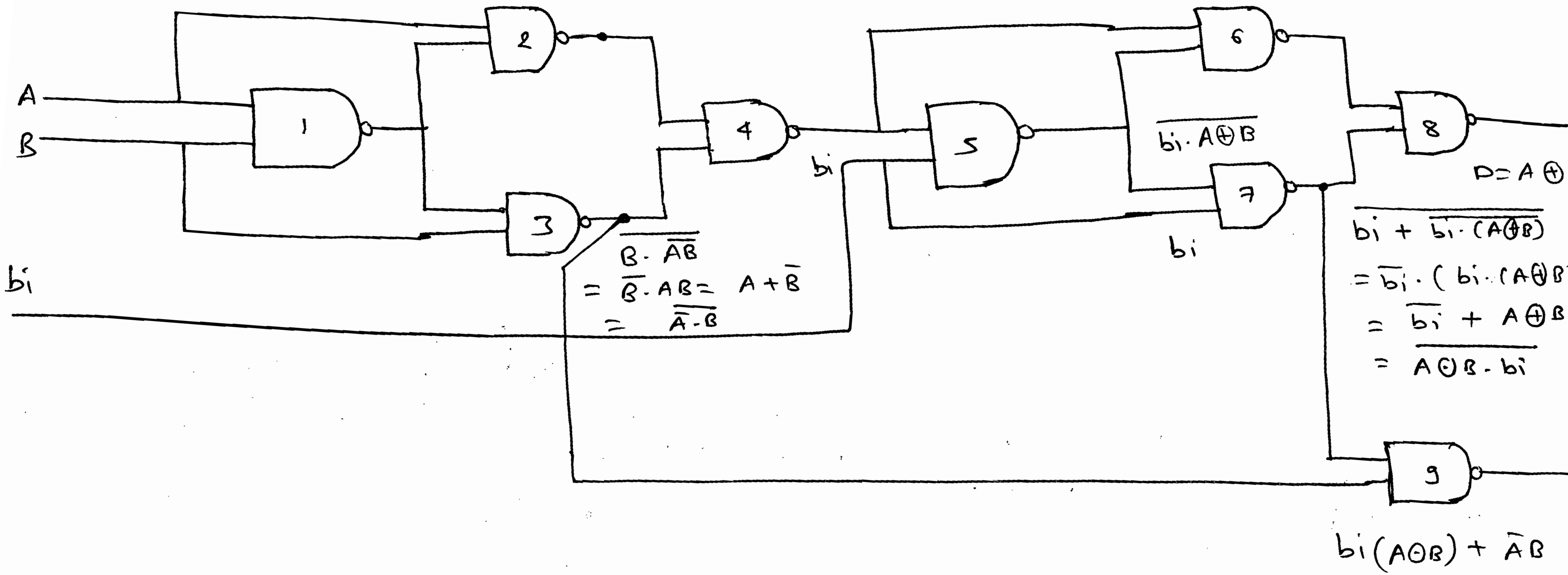
$D = A \oplus B \oplus bi$

$B = bi\,(A \odot B) + \bar{A}B$

* Full Subtractor using NAND gate only:

$\overline{A \cdot \overline{AB}} = \bar{A} + AB = \bar{A}+B$

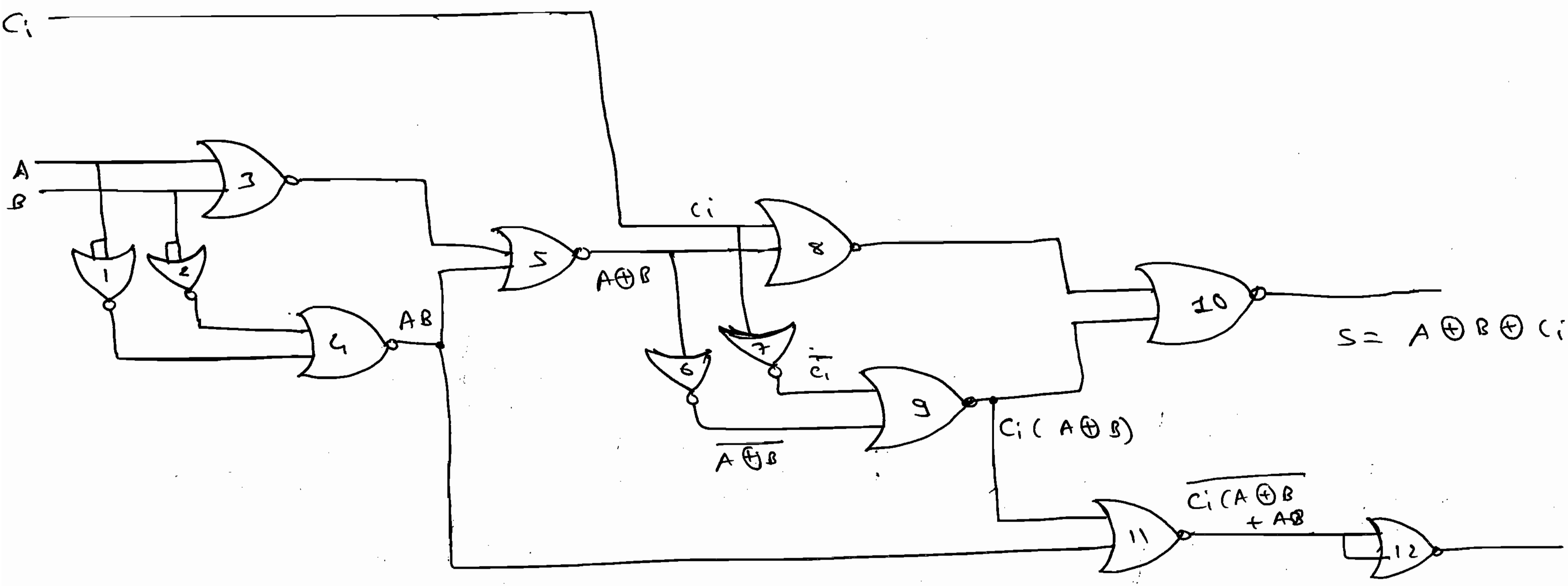
$= \overline{A\bar{B}}$



$D = A \oplus B \oplus b_i$

$B = bi\,(A \odot B) + \bar{A}B$

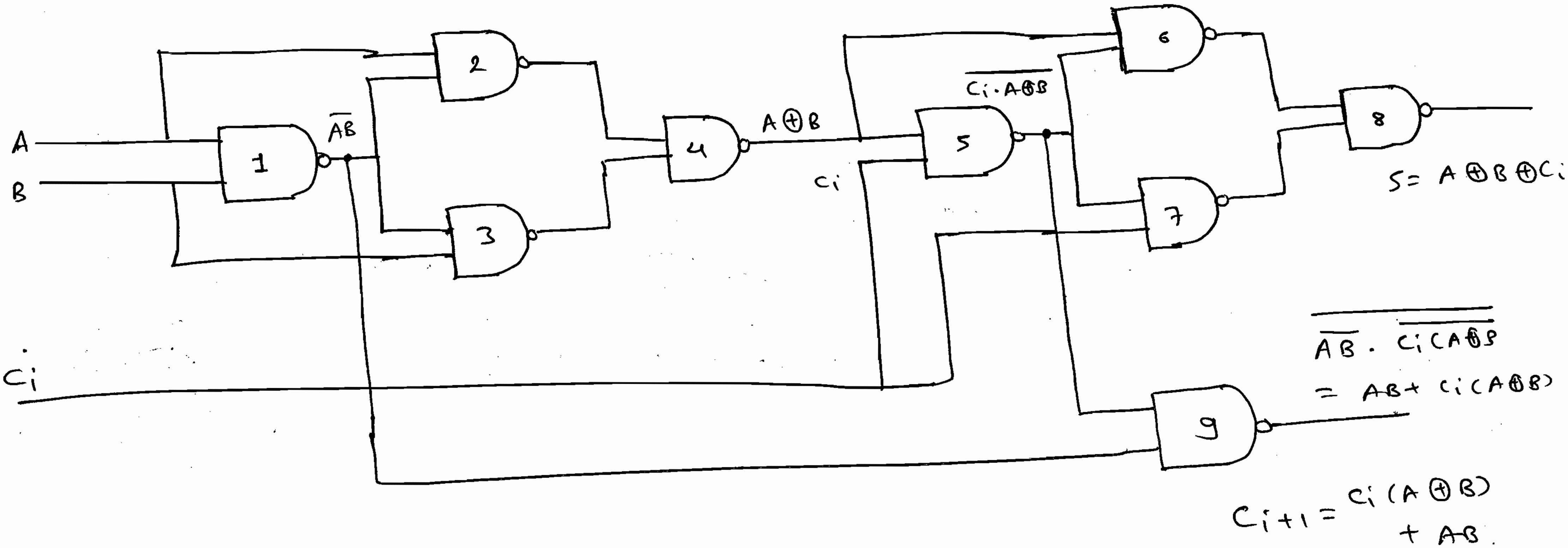
* Full Adder using NOR gate only:



$S = A \oplus B \oplus C_i$

$C_{i+1} = C_i(A \oplus B) + AB$

* Full adder using NAND gate only



$S = A \oplus B \oplus C_i$

$C = C_i \cdot (A \oplus B) + AB$